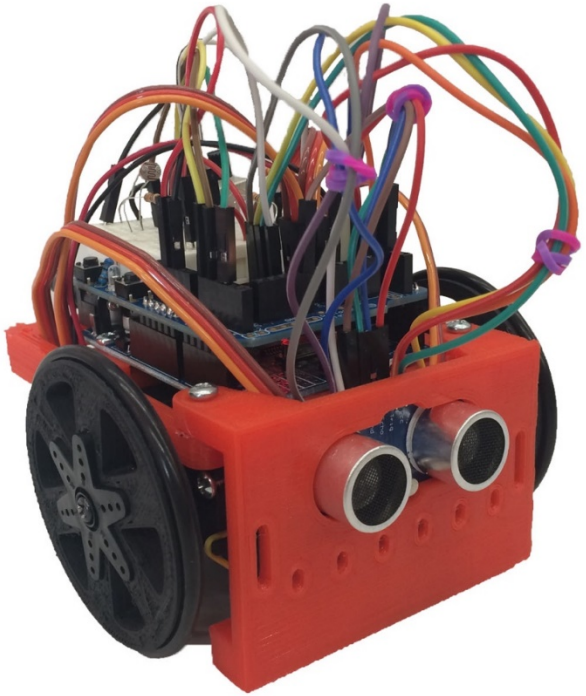
TuBot 2017

GUIA DE MONTAJE Y PROGRAMACIÓN

Introducción a la programación

Sensores



Sistema Basado en Microprocesador

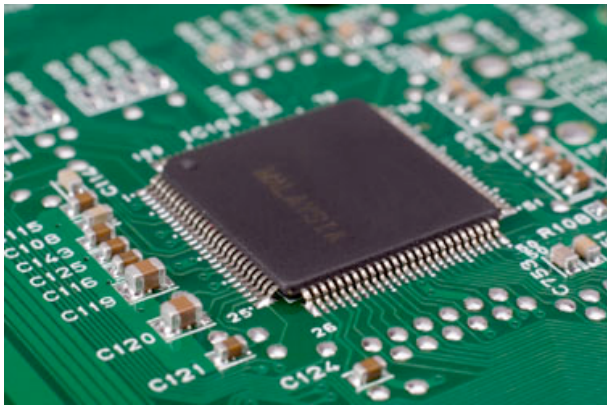
- El **Microprocesador** es el cerebro de un ordenador
- Para funcionar necesita un **Programa** donde se indican las **órdenes** que el procesador debe seguir.
- **Elementos** de un Sistema basado en Microprocesador:
  - CPU (Central Processing Unit)
  - Memoria
  - Periféricos

Source: http://www.callegranvia.com/images/product/375/1b78d5c0b624c9a3c966e0854829dfe6.jpg

Programa y compilador

- Un **Programa** se puede escribir en muchos **lenguajes** diferentes.

- El microprocesador realmente entiende sólo órdenes en **código máquina** (**unos y ceros**)



Source: http://pacotraver.files.wordpress.com/2011/11/interprete.jpg
http://2.bp.blogspot.com/_Pm8qvnCsVOI/TCwv_SAuznI/AAAAAAAAAA4/9asQgJGiQMw/s1600/MICRO.jpg

Programa y compilador

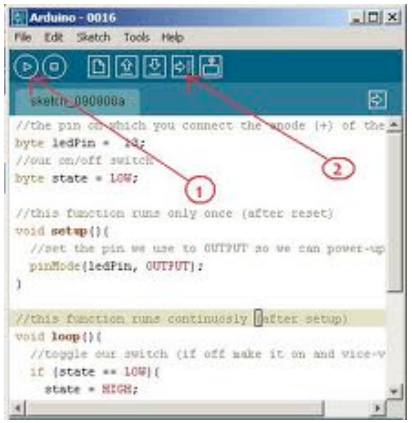
- Pero para nosotros es un poco "complicado" y nos es más fácil utilizar el **inglés** (como lengua internacionalmente más extendida)
- Nosotros vamos a utilizar una versión simplificada del **lenguaje C++** que es un **lenguaje de programación en alto nivel**
  - Las órdenes básicas están en **Inglés**
  - Tiene pocas **reglas gramaticales** pero muy estrictas.
- Un **compilador** pasa las órdenes del lenguaje de alto nivel al código máquina que entiende el procesador

Source: http://www.microchip.com/stellent/images/mchpsiteimages/c32_CoverMainGfx.jpg

Arduino: buen sistema de iniciación

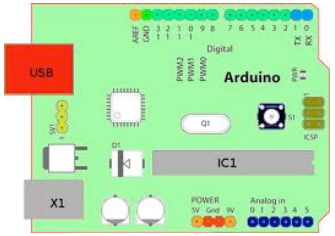
- Plataforma diseñada como elemento de **iniciación a la programación y a la electrónica**.
- Muy **sencillo** de utilizar.
- Dispone de una enorme **comunidad de usuarios**.

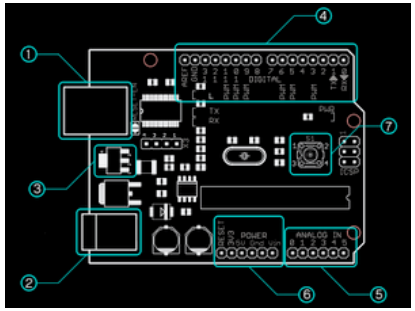
ARDUINO



Arduino: buen sistema de iniciación

- Desarrollada totalmente como **open-source.**
  - Los **esquemas** de las tarjetas hardware están disponibles para poderlos replicar.
  - Las fuentes de las **librerías** están disponibles para poder modificarlas y utilizarlas.
  - El **entorno de programación** también es abierto



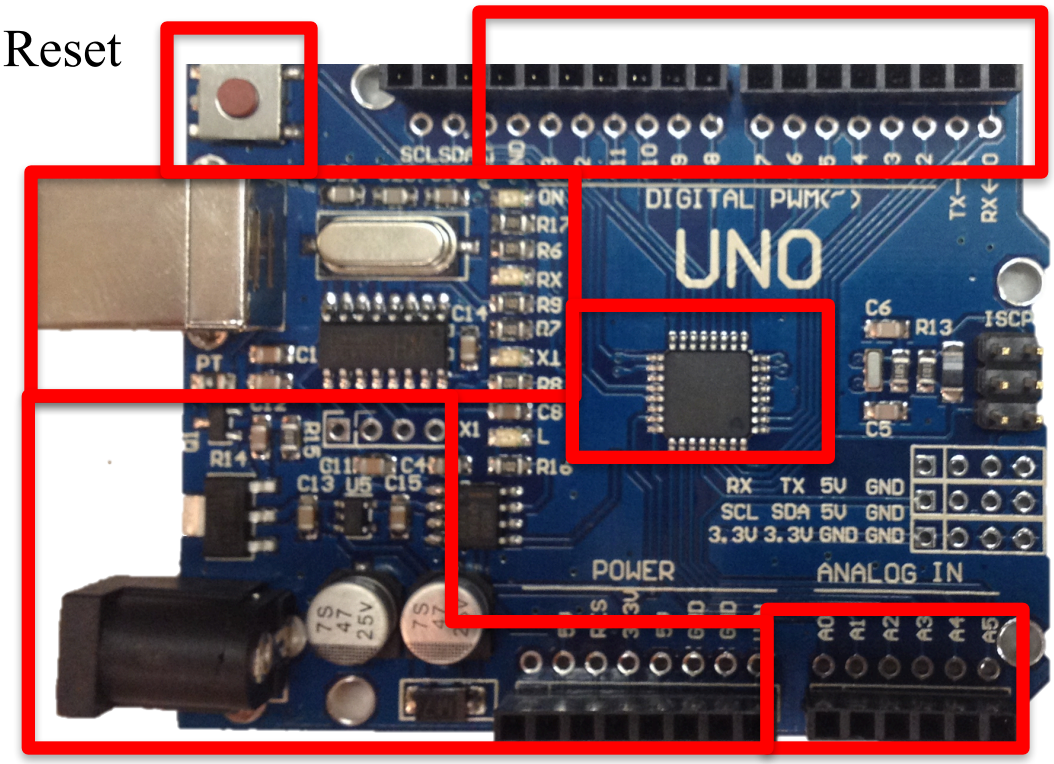


- **Multiplataforma**: Windows, Linux, MAC OS-X

Source: http://www.ikkaro.com/definicion-arduino http://proyectos-arduino-geometriaemocional.blogspot.com.es/2012/03/jugando-con-sonido.html

Arduino: buen sistema de iniciación

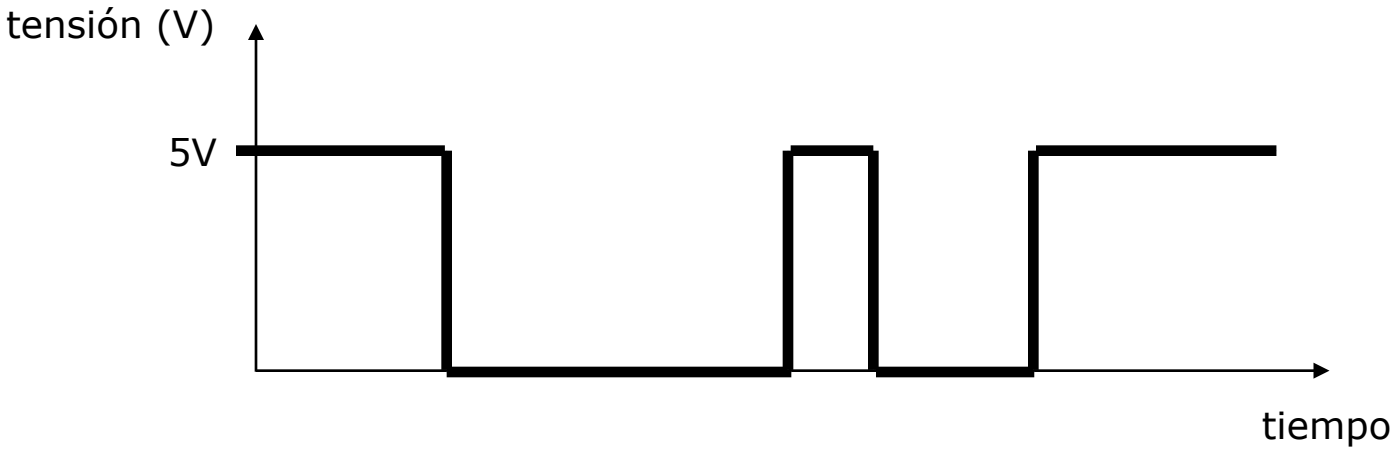
- Elementos de la tarjeta **Arduino Uno**



Entradas y Salidas Digitales

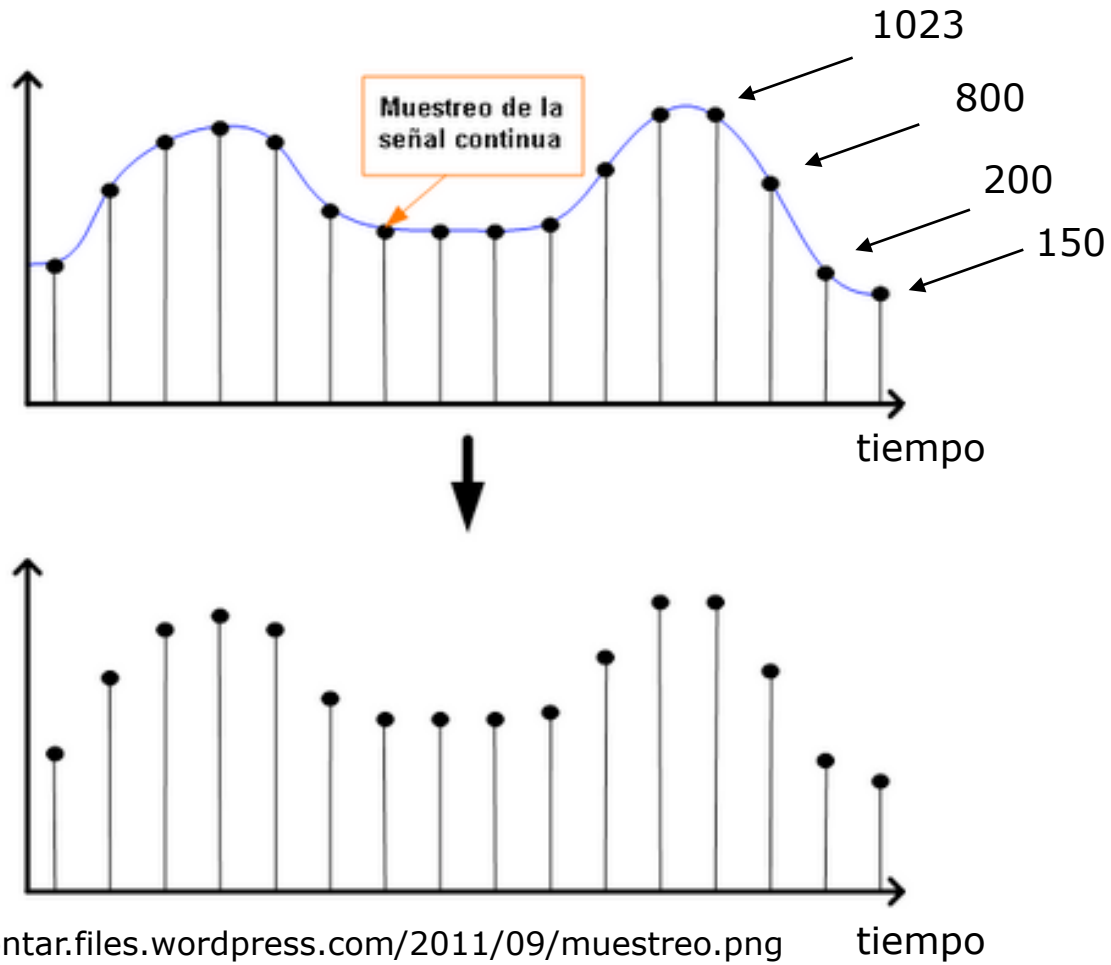
‘1’ → 5 voltios → Nivel Alto → HIGH

‘0’ → 0 voltios → Nivel Bajo → LOW



Entradas Analógicas

- La tensión varía de 0 voltios a 5 voltios



Source: http://oirverycontar.files.wordpress.com/2011/09/muestreo.png

Arduino: buen sistema de iniciación

- Entradas y salidas de la tarjeta Arduino Uno y compatibles

GND = 0V

Instalando el software en el ordenador

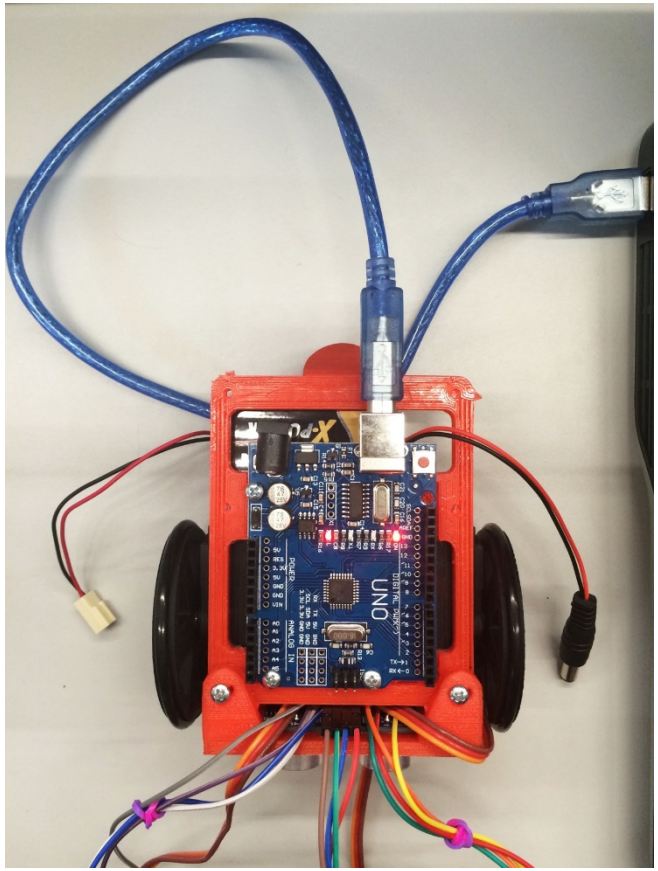
- **Actividad**
  - Si en el ordenador no están instaladas las herramientas de Arduino se deben descargar.
    - Descargar el entorno de programación de Arduino de http://arduino.cc/en/pmwiki.php?n=main/software
      - Nosotros tenemos instalada la versión 1.8.5
    - Seguir las instrucciones proporcionadas por Arduino:
      - En inglés: http://arduino.cc/en/Guide/Windows
      - En castellano: http://arduino.cc/es/Guide/Windows#
  - Resumen de instrucciones:
    - Descomprimir el fichero descargado en el directorio donde se quiera tener la aplicación
    - Conectar la tarjeta al ordenador con el cable USB
    - Instalar el Driver
      - Para Windows 10 y versiones anteriores.
        - http://www.wch.cn/downloads/downfile.php?id=65
    - Ejecutar la aplicación Arduino

Cargando mi primer programa

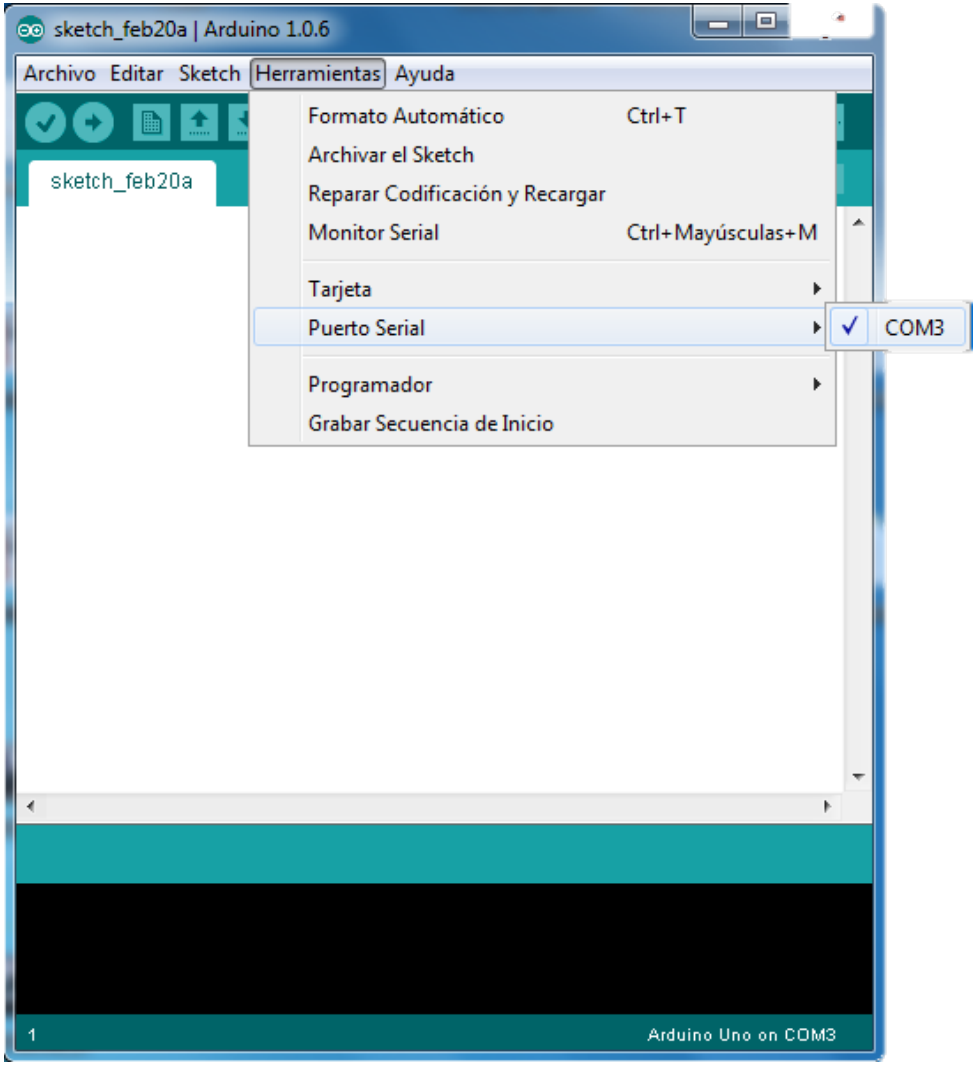
Actividad

- Conecta la tarjeta al ordenador por USB

Coge el cable USB de la caja y conéctalo al ordenador como en la foto.



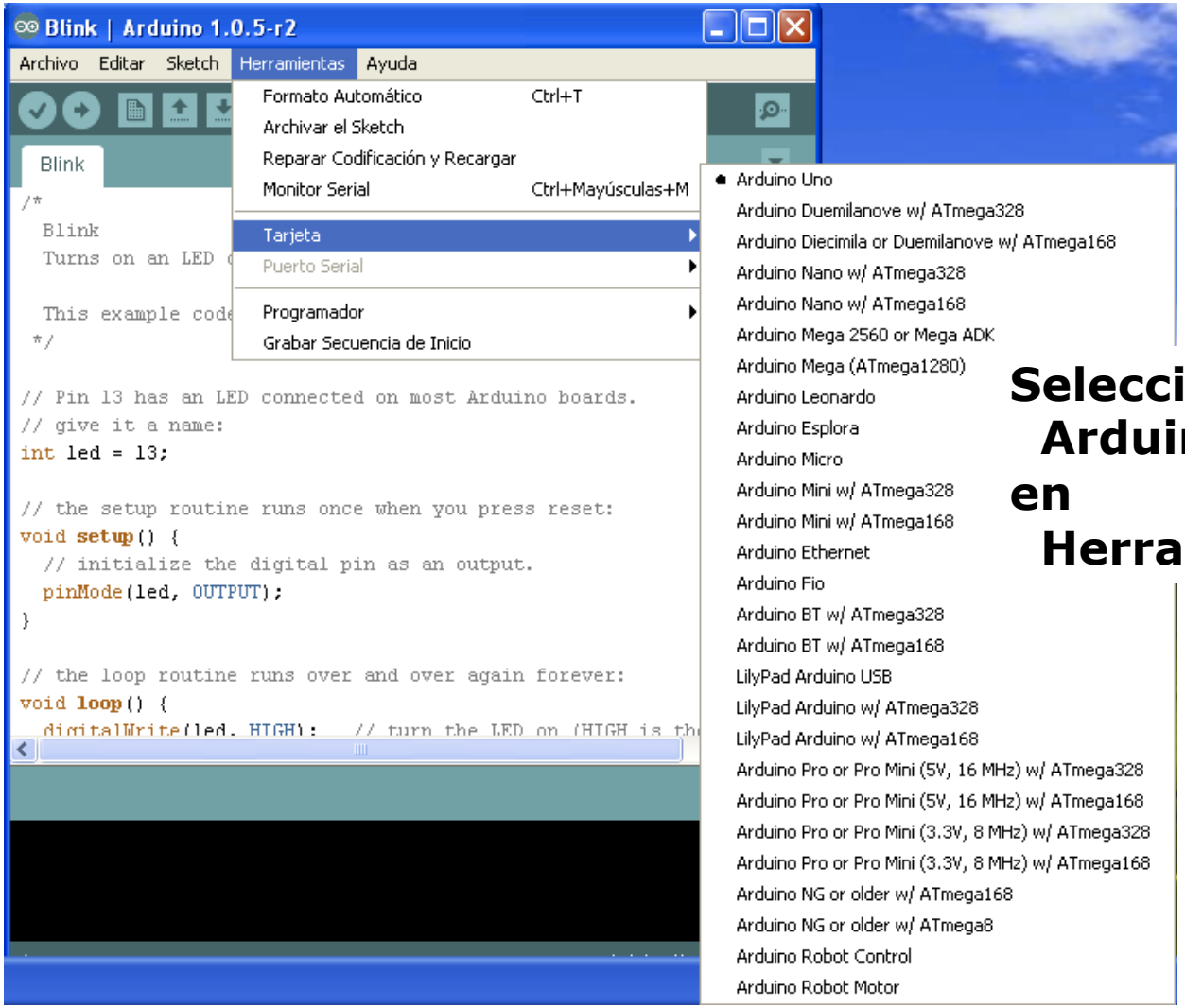
Cargando mi primer programa



Verifica que el Puerto Serial coincide con el puerto que está conectada la tarjeta.

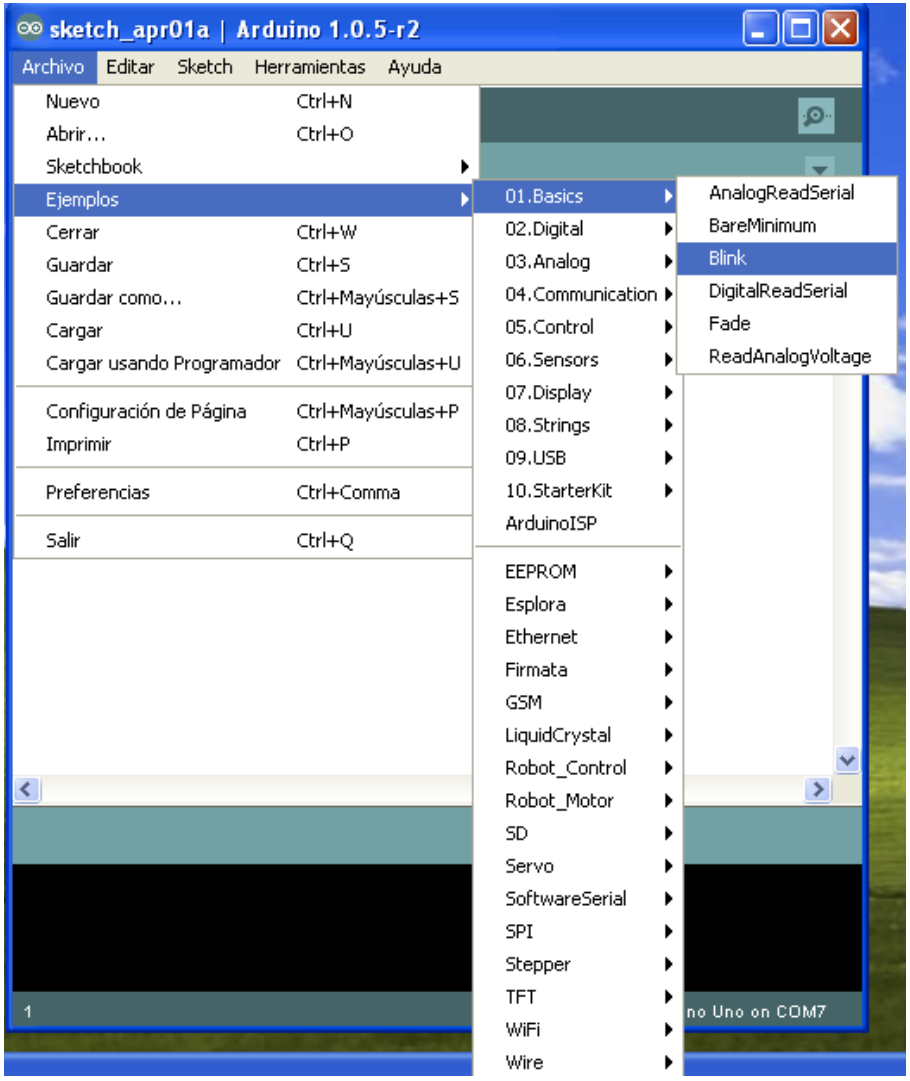
Para evitar problemas es importante que la tarjeta esté conectada por USB antes de abrir el entorno de Arduino

Cargando mi primer programa



Selecciona
Arduino UNO
en
Herramientas -> Tarjeta

Cargando mi primer programa



Verificar que el programa es correcto

Cargar el programa verificado en la placa

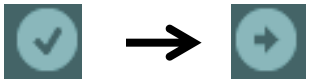
Crear un nuevo programa

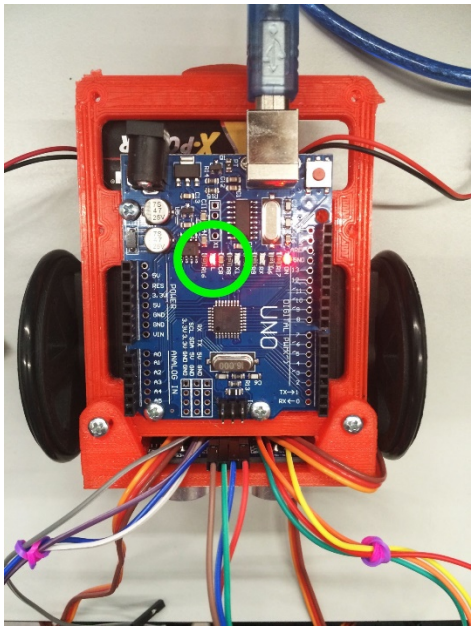
Abrir un programa existente

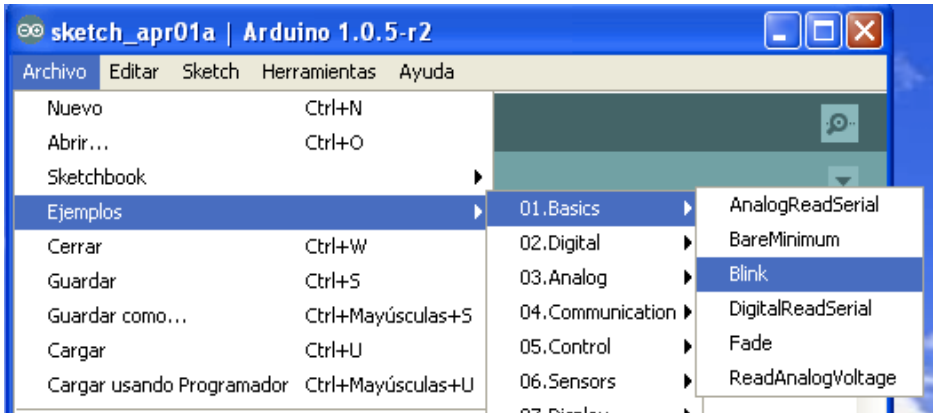
Guardar las modificaciones del programa

Cargando mi primer programa

Actividad

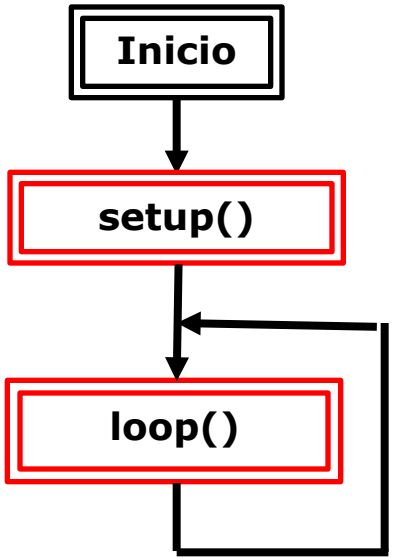
- Carga el ejemplo “Blink” (parpadeo)
  - Archivo → Ejemplos →0.1 Basics→ Blink
- Verifica y descarga el programa →  →
- Comprueba que el LED de la tarjeta parpadea.





Entendiendo mi primer programa

- Estructura básica de un programa de Arduino
  - Todos los programas de Arduino tienen dos partes:
  - setup()
    - Se ejecuta sólo una vez al principio (al encenderlo o al cargarlo)
    - Se utiliza para configurar las entradas y salidas, para inicializar variables, …
  - loop()
    - Se ejecuta continuamente sin fin




```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fíjate en la sintaxis (llaves y paréntesis)

Entendiendo mi primer programa

- Ejemplo “Blink” (“Parpadeo”)
  - A continuación te explicamos el programa que has cargado

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Entendiendo mi primer programa

- Ejemplo “Blink” (“Parpadeo”)
  - Se identifican claramente el “setup()” y el “loop()”

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Entendiendo mi primer programa

- Ejemplo “Blink” (“Parpadeo”)
  - Se identifican claramente el “setup()” y el “loop()”

```
// Pin 13 has an LED connected
// give it a name:
int led = 13;

// the setup routine runs once
void setup() {
  // initialize the digital pi
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Esto es una variable:

- Es un elemento del programa que almacena datos (es un contenedor de datos)
- Los datos almacenados pueden cambiar
- Hay variables de diferentes tipos

- En este caso se está creando una variable llamada “led” de tipo “int” a la que se asigna el número 13

Entendiendo mi primer programa

- Ejemplo "Blink" ("Parpadeo")
  - Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected
// give it a name:
int led = 13;

// the setup routine runs once
void setup() {
  // initialize the digital pi
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Esta es la orden para configurar un pin digital

- **pinMode** es una función con dos parámetros
  - El primer parámetro es el número de la entrada o salida digital a configurar
  - El segundo parámetro es:
    - OUTPUT → Para configurarlo como salida
    - INPUT → Para configurarlo como entrada
- Esta función configura el pin 13 como salida.

Entendiendo mi primer programa

- Ejemplo “Blink” (“Parpadeo”)
  - Se identifican claramente el “setup()” y el “loop()”

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pi
  pinMode(led, OUTPUT);
}

// the loop routine runs over
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

Estas ordenes cambian el valor de una salida digital

- **digitalWrite** esta función tiene dos parámetros:
  - El primer parámetro es el número de la salida digital que se quiere modificar
  - El segundo parámetro es:
    - HIGH→ Para ponerla a nivel alto (‘1’ ó 5V)
    - LOW → Para ponerla a nivel bajo (‘0’ ó 0V)
- Estas funciones actúan sobre la salida digital 13 poniéndola primero a nivel alto (5V que encienden el LED) y luego a nivel bajo (0V que apaga el LED).

Entendiendo mi primer programa

- Ejemplo "Blink" ("Parpadeo")
  - Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

Esta orden introduce una pequeña espera

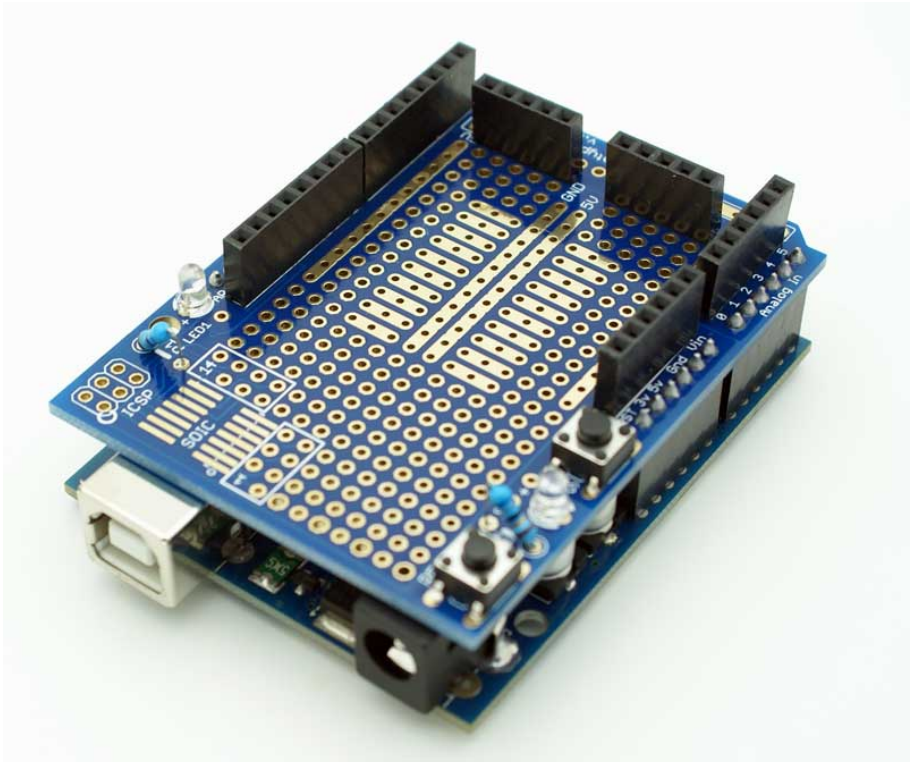
- **Delay** esta función tiene un parámetro:
  - Como parámetro se introduce el tiempo de espera en milisegundos
    - 1000 → 1000ms = 1 segundo

Entendiendo mi primer programa

- **Ejercicios**:
  1. Modifica el programa para que el LED parpadee más rápido
     - Escribe, verifica el programa y cárgalo en la placa
  2. Modifica el programa para que repita una secuencia de dos parpadeos rápidos consecutivos seguidos de un tiempo de apagado
     - Escribe, verifica el programa y cárgalo en la placa

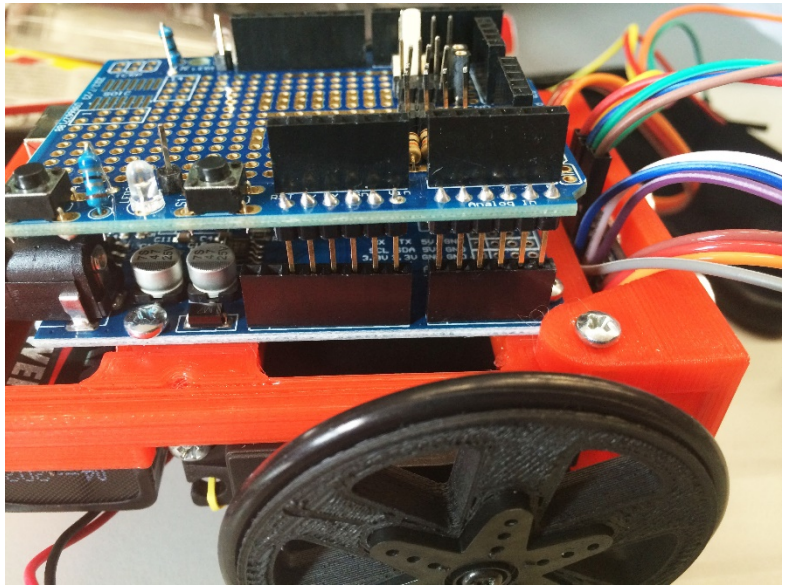
Tarjeta de conexiones

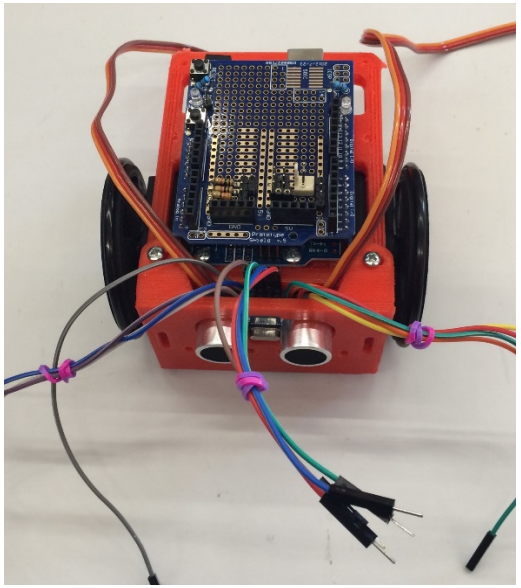
- Conecta la tarjeta de conexiones sobre la tarjeta de control



Preparación de la tarjeta de conexiones

- Conecta la tarjeta de conexiones sobre la tarjeta de control

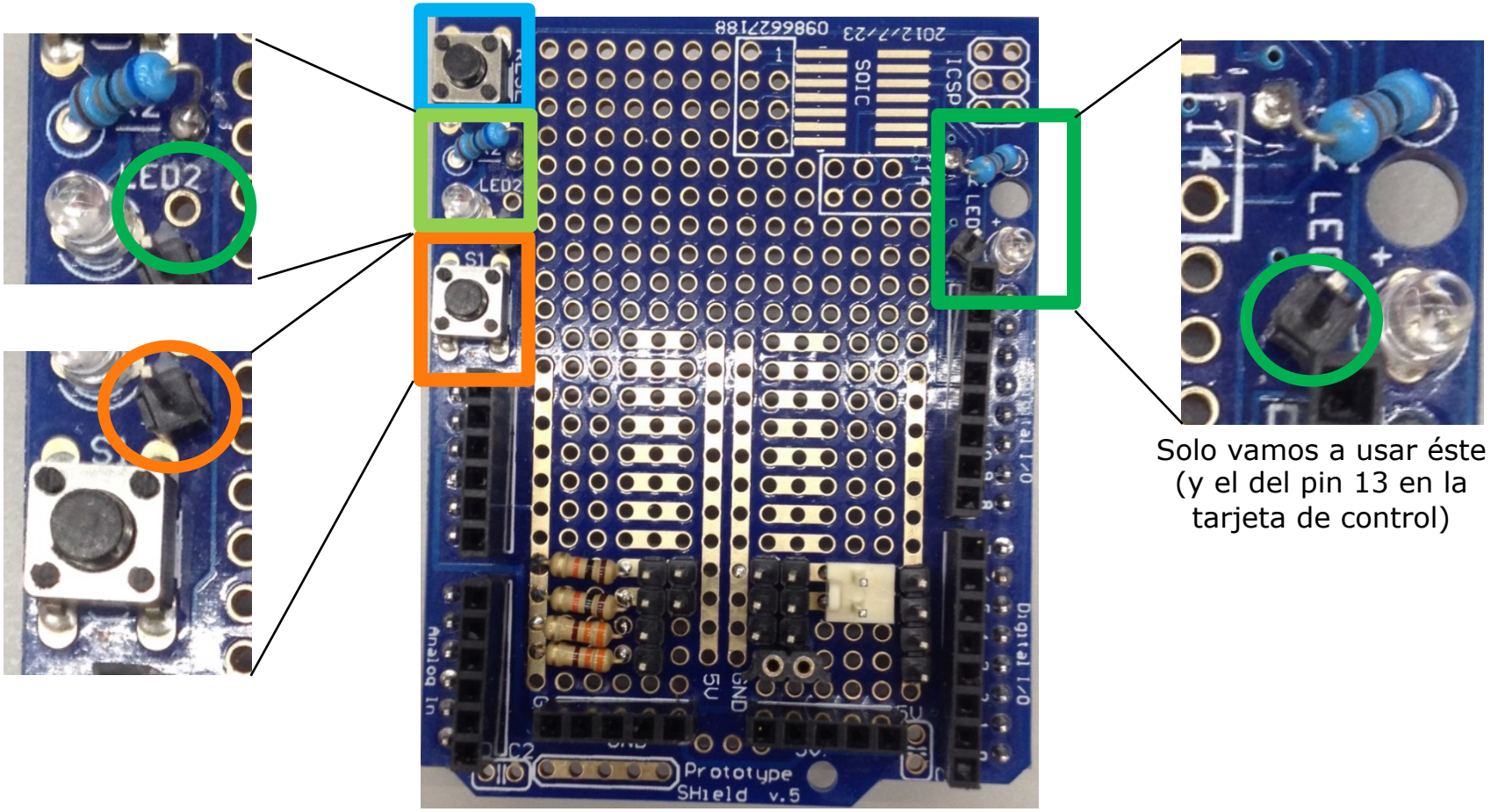




- Con ella tendremos un pulsador, dos LEDs y varias resistencias y pines que nos hará falta más adelante
- Cuidado al poner la tarjeta de conexiones para no doblar los pines.
- Asegúrate de que coinciden.

Preparación de la tarjeta de conexiones

- En la tarjeta está integrado el **Reset**, dos **LEDs** y un **pulsador**

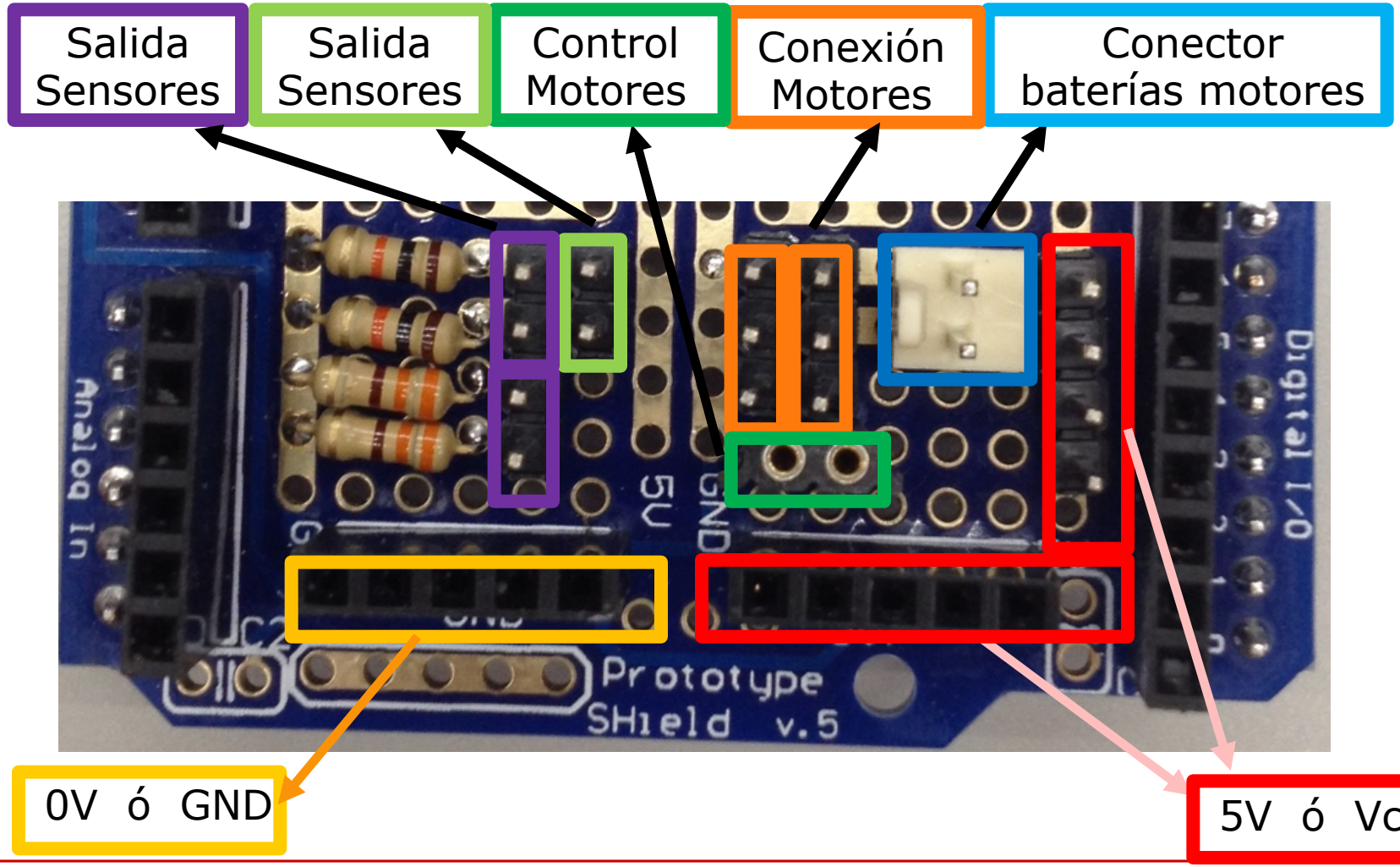


Solo vamos a usar éste (y el del pin 13 en la tarjeta de control)

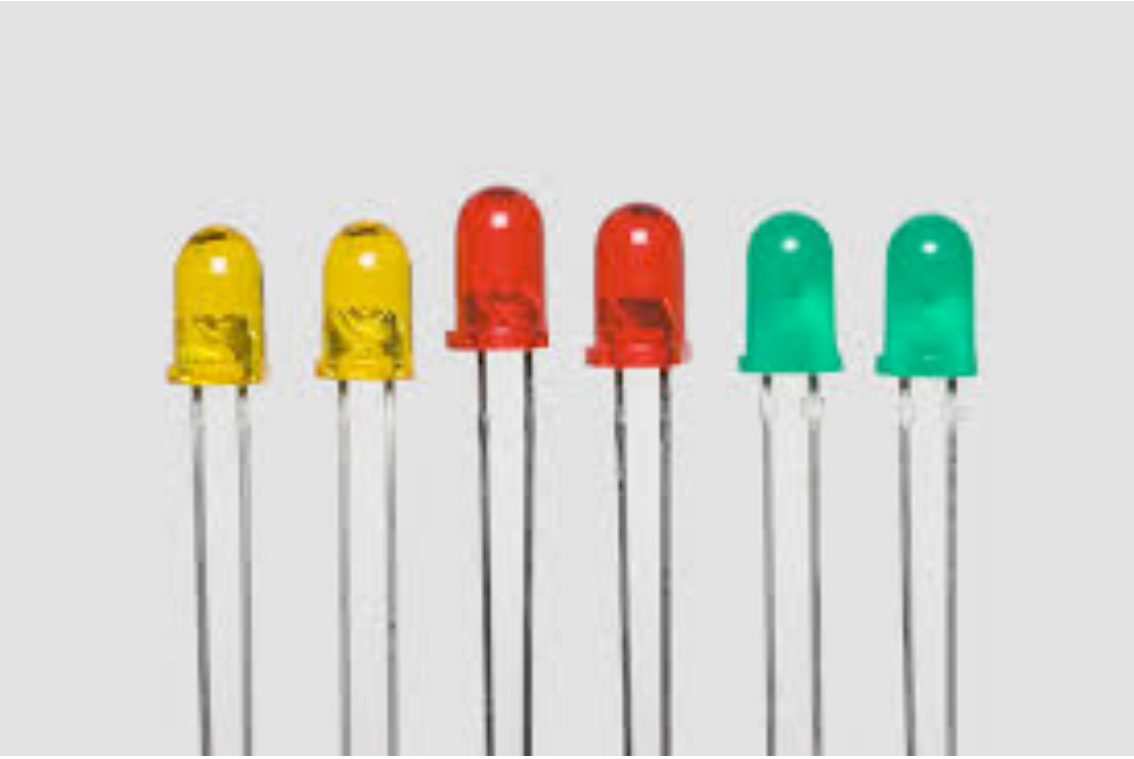
- También hay soldados varios componentes que se verán más adelante

Conexiones de la tarjeta de conexiones

- Detalle de todas las conexiones de la tarjeta.
  - No te asustes, irás conectando las cosas poco a poco.



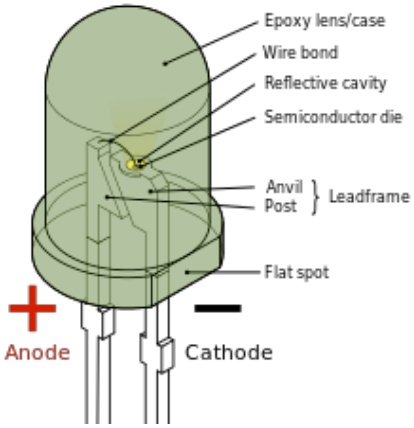
Entendiendo el funcionamiento de un LED

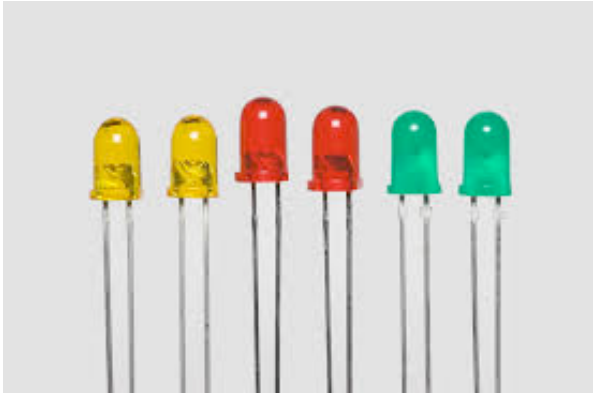


Source: http://en.wikipedia.org/wiki/Light-emitting_diode
http://www.dreamstime.com/royalty-free-stock-image-set-color-leds-image26386376
http://www.ecotownstore.com/led/flood/transportation.html

Entendiendo el funcionamiento de un LED

- Un LED (Light Emiting Diode – Diodo emisor de luz)
  - Los LEDs los vemos constantemente en los aparatos electrónicos
  - Dispositivo que deja pasar la corriente en un sentido (en el otro no)
  - Cuanta más corriente pasa más luce.

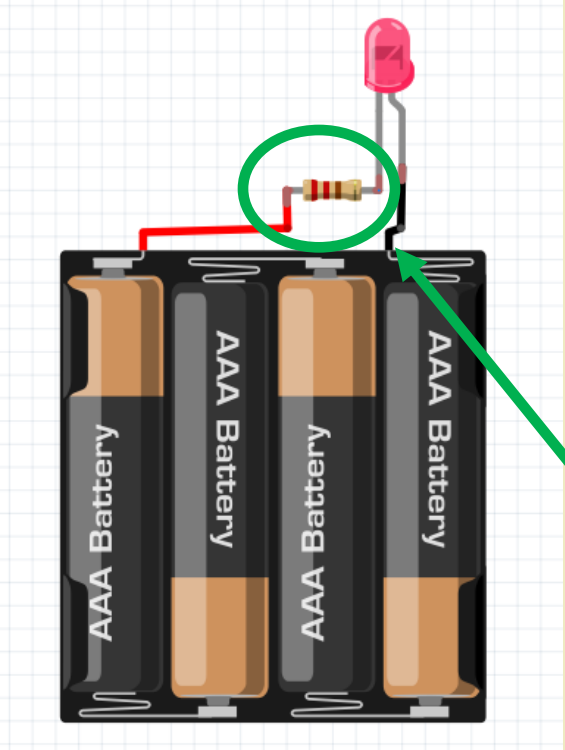


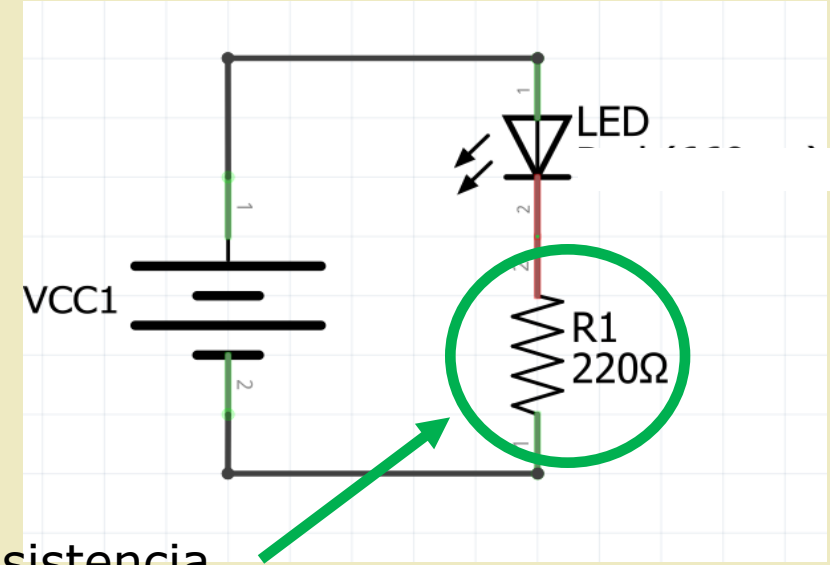


Source: http://en.wikipedia.org/wiki/Light-emitting_diode
http://www.dreamstime.com/royalty-free-stock-image-set-color-leds-image26386376
http://www.ecotownstore.com/led/flood/transportation.html

Entendiendo el funcionamiento de un LED

- El circuito eléctrico del LED
  - Cuando se conecta un LED es necesario que haya una resistencia que limite la corriente que pasa por él (si no se pone se quema).
  - Cuanto mayor es el valor de la resistencia, menos corriente pasa y menos luce el LED

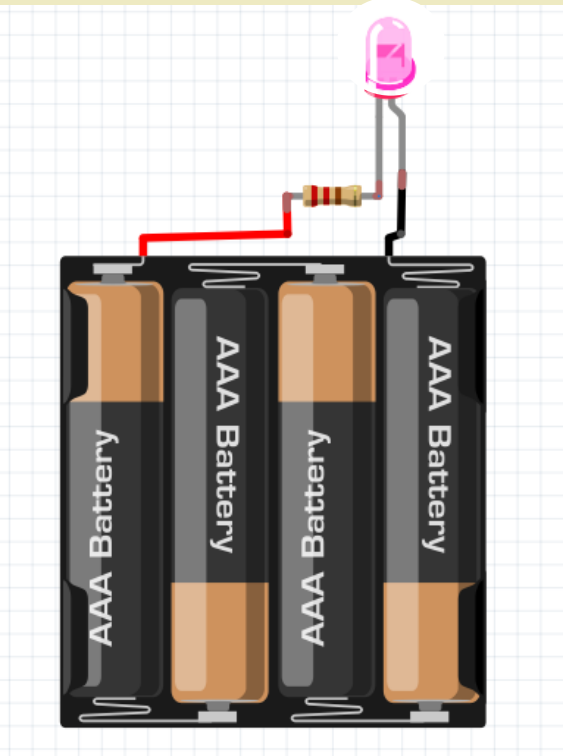


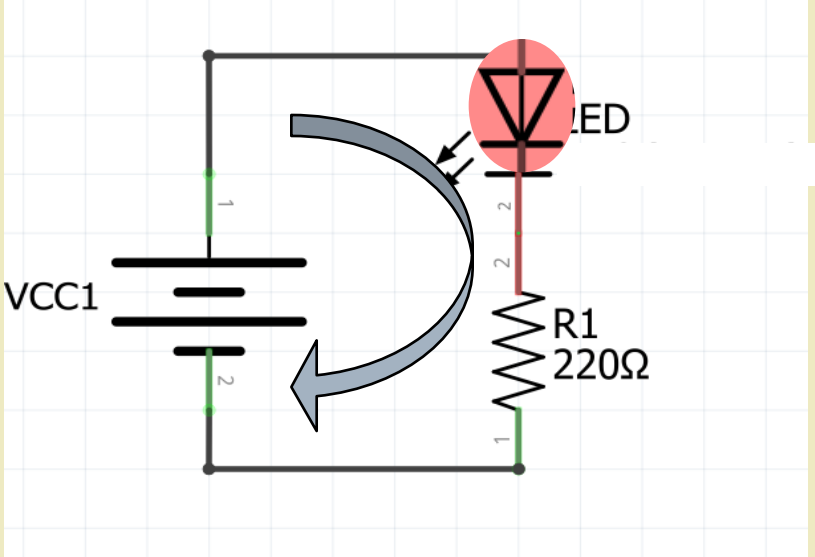


Resistencia

Entendiendo el funcionamiento de un LED

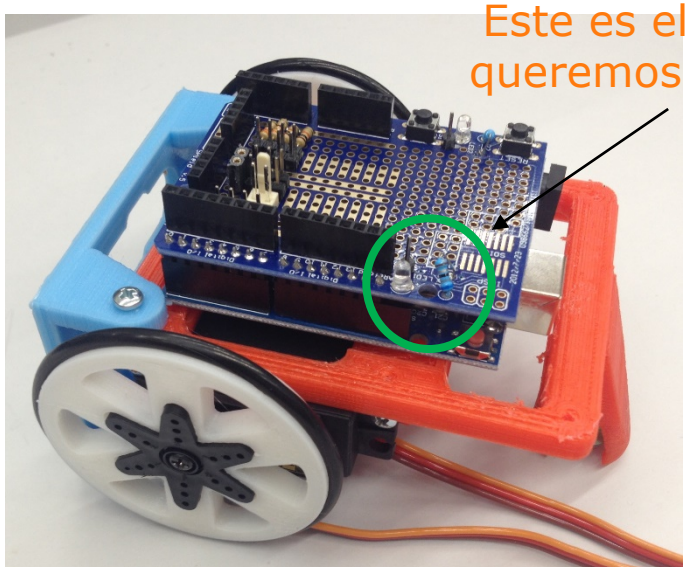
- El circuito eléctrico del LED
  - Cuando se cierra un circuito eléctrico, circula una corriente del polo positivo al negativo de la batería.
  - El LED tiene polaridad. Sólo luce cuando pasa corriente por él, y sólo pasa corriente en un sentido.





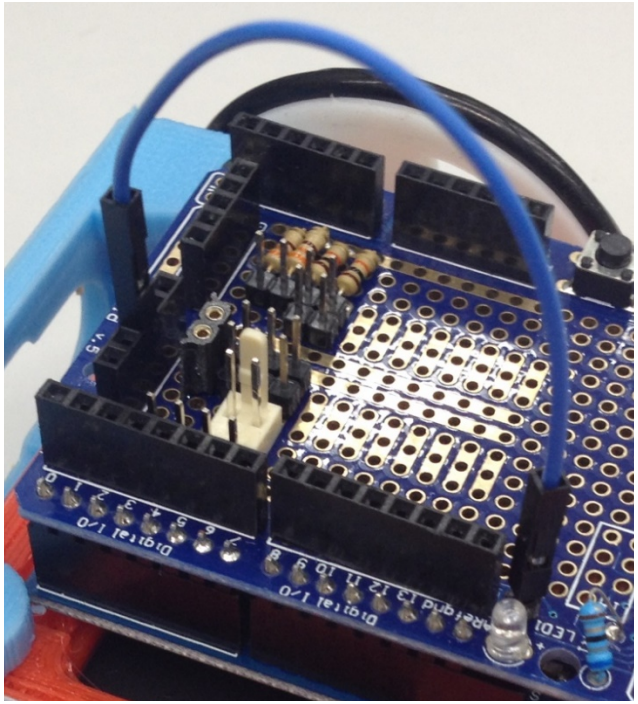
Entendiendo el funcionamiento de un LED

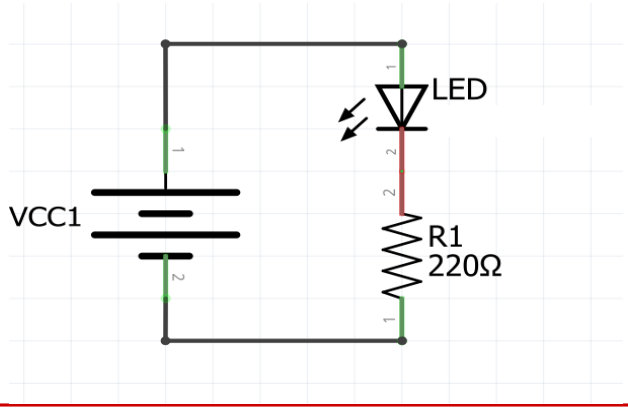
- En esta tarjeta hay un LED que ponemos conectar



Conectamos el cable azul a la línea de 5V tal como indica la foto.

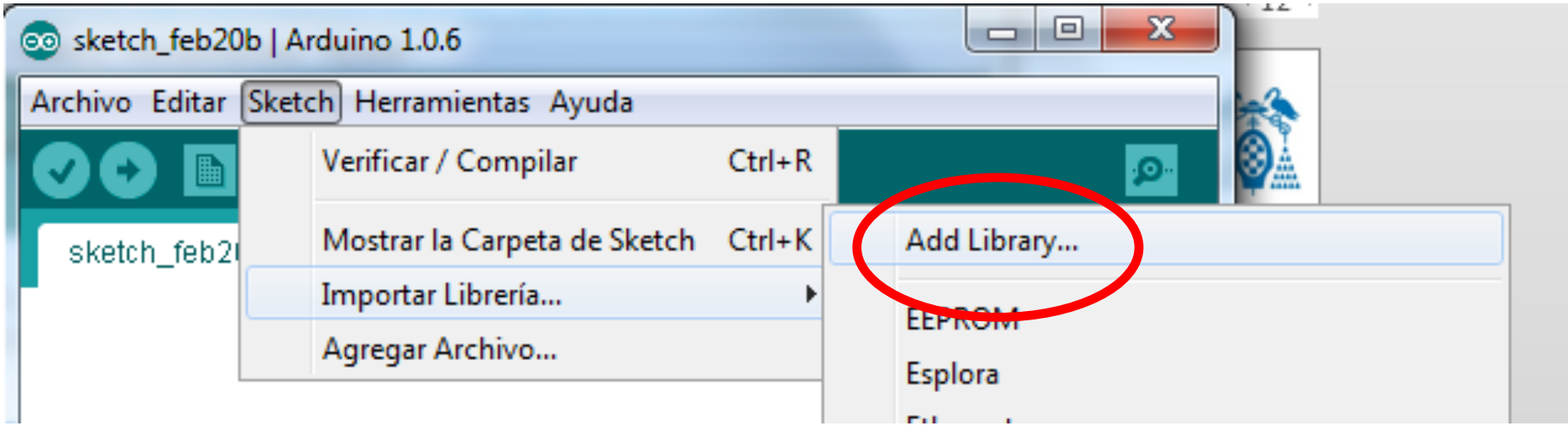
Si la tarjeta está conectada al ordenador debería encenderse el LED y apagarse al desconectarlo





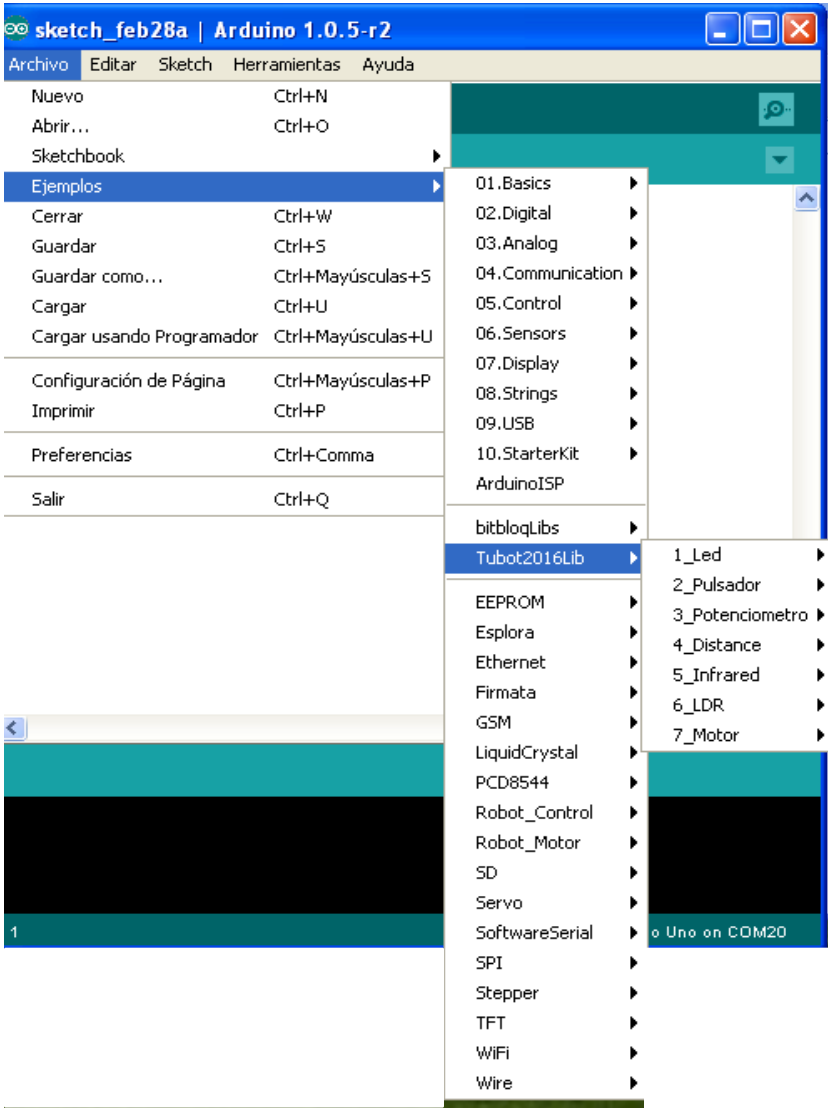
Instalando las librerías

- Importar librerías (Bibliotecas de funciones):
  - Una librería es un "conjunto de comandos o funciones" y ejemplos.
  - Para poder usar una librería primero debemos importarla
  - La librería que se debe importar se llama "Tubot2016Lib"
    1. Busca el fichero **Tubot2016Lib.zip** o descárgalo de la página web de TuBot
    2. Para importar una librería nos vamos al menú de Arduino y seleccionamos Sketch > Importar Librería > Add Library
    3. Selecciona el fichero **Tubot2016Lib.zip** y selecciona **abrir**



Instalando las librerías

- Importar librerías:
  - Una vez hecho esto ya tenemos la librería importada y la podemos utilizar incluyendo en nuestro proyecto el archivo:
    - #include <Tubot2016Lib.h>
  - Cada librería incluye además una serie de ejemplos a los que podemos accedemos a través del menú de Arduino.
  - Añade también la librería "DistanceSRF04" para el sensor de ultrasonidos.

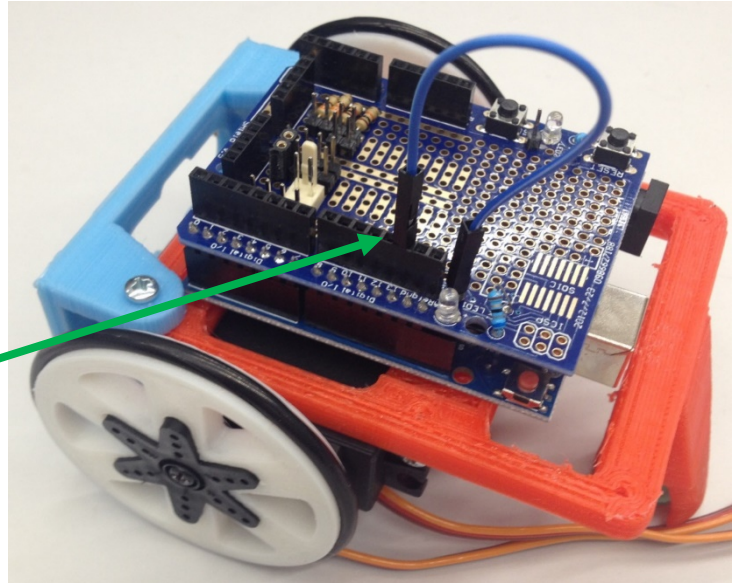


Conexión de un LED al PIN 13

- Ahora vamos a conectar el led a la tarjeta de control.
- Pasos:
  - Paso 1: Conectar el LED al PIN 13 en vez de a 5V
  - Paso 2: Cargar el programa "ParpadeaLed"
    - Archivo->Ejemplos->Tubot2016Lib->1_Led->ParpadeaLed
    - El ejemplo "ParpadeaLed" es muy parecido al "Blink" que has visto antes.
  - Resultado:
    - El LED debe parpadear

Pin13

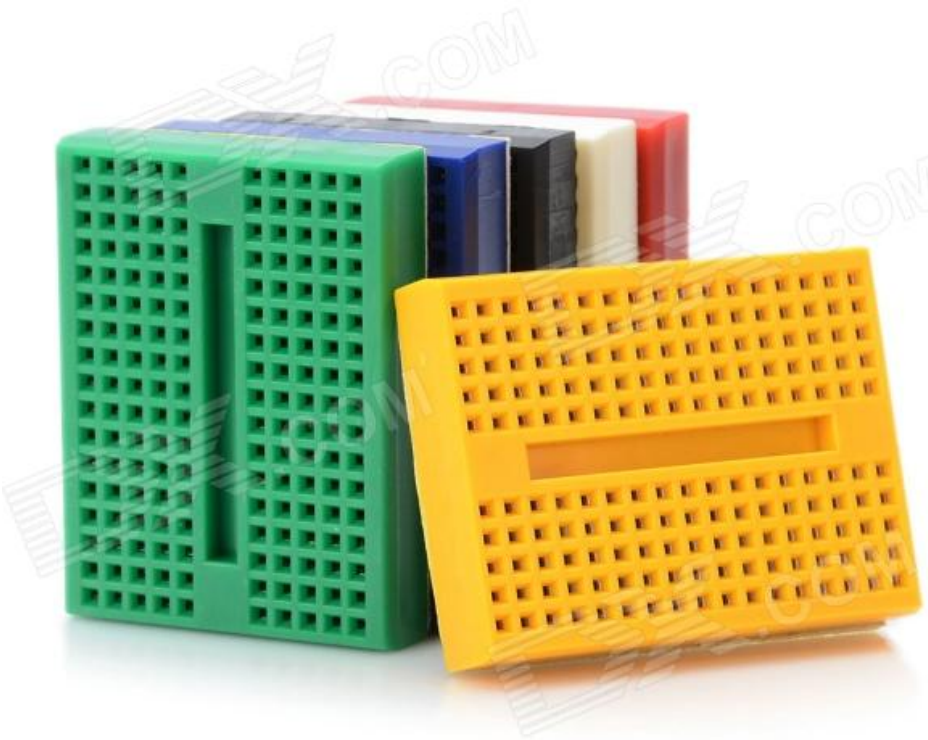
Este cable **Macho – Hembra** puede ser azul o morado



Seguimos utilizando el LED

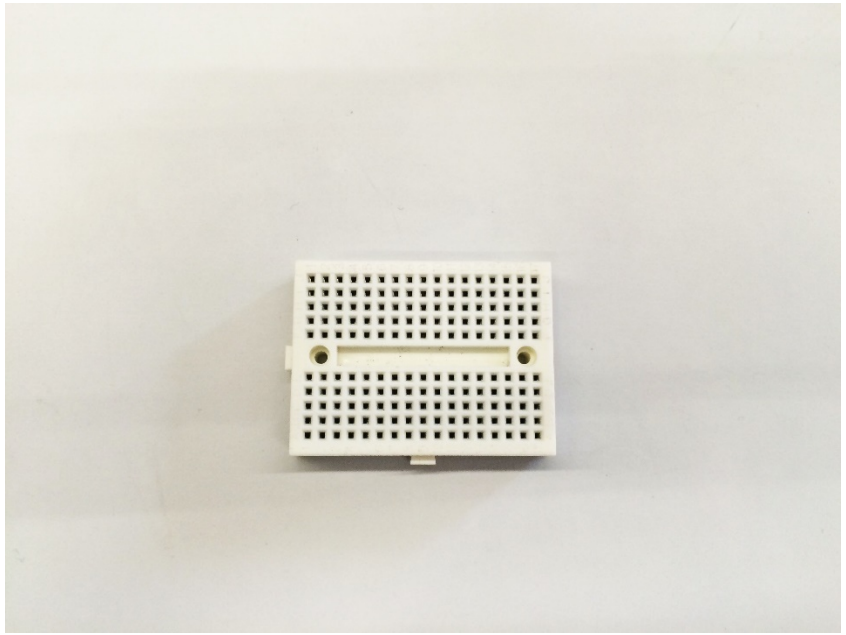
- **Ejercicios**:
  1. Conecta el LED al pin 7 y cambia el programa ParpadeaLED para que parpadee este LED en vez del del PIN 13.
     - Escribe, verifica el programa y cárgalo en la placa
  2. Modifica el programa para que parpadeen los LEDs conectados al pin 13 (en la tarjeta de abajo) y el que se ha conectado al pin 7 de manera que cuando uno se apague se encienda el otro y viceversa.
     - Escribe, verifica el programa y cárgalo en la placa
  3. Modifica el programa para que parpadeen los dos LEDs pero el LED del pin 13 debe hacerlo cuatro veces más rápido que el del pin 7.
     - Escribe, verifica el programa y cárgalo en la placa

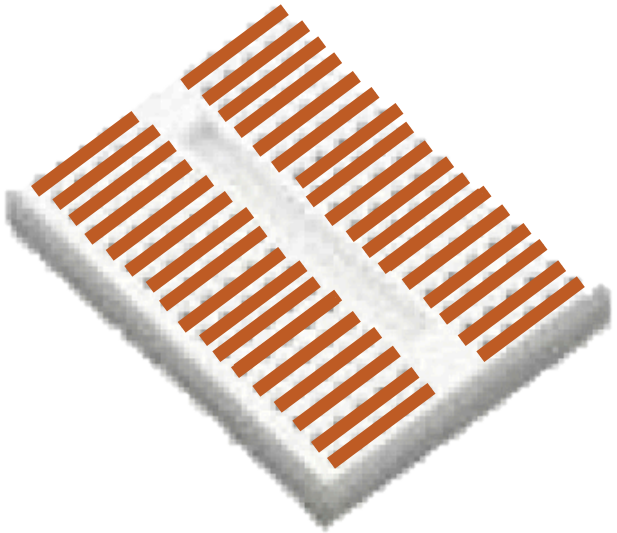
Placa de prototipado rápido



Preparación de la tarjeta de conexiones

- Placa de prototipado rápido (BreadBoard o ProtoBoard)
  - Sobre la tarjeta de conexiones se coloca una placa de prototipado rápido para poder realizar conexiones más fácilmente.
  - Utilizada para hacer conexiones eléctricas sencillas.

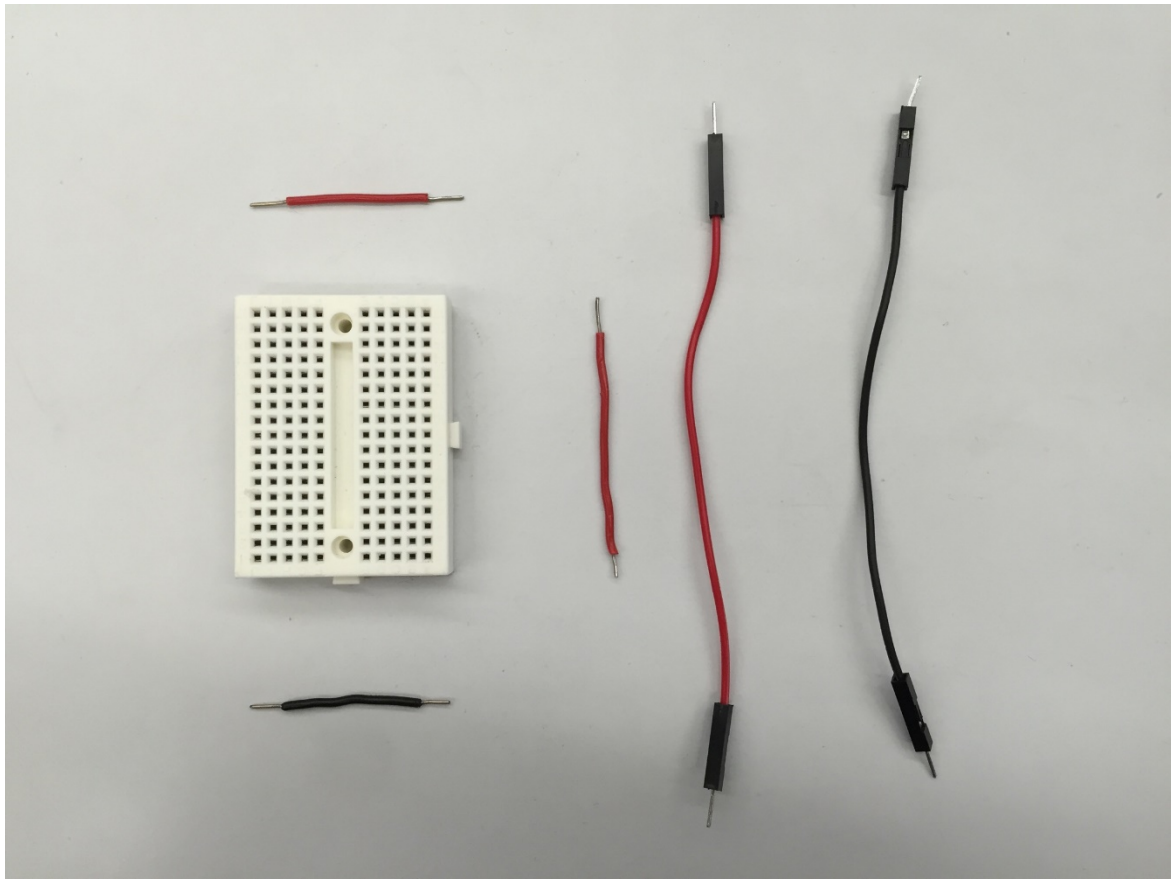




  - Las filas de 5 cuadros de los laterales están conectadas internamente entre sí

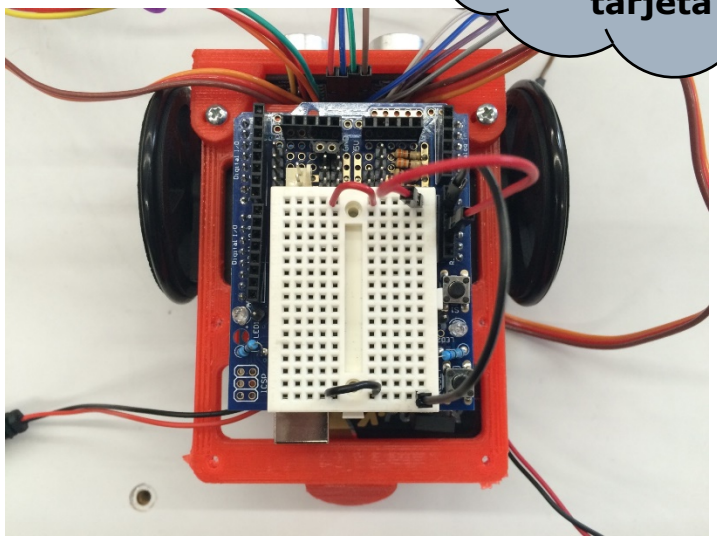
Preparación de la ProtoBoard

- Materiales

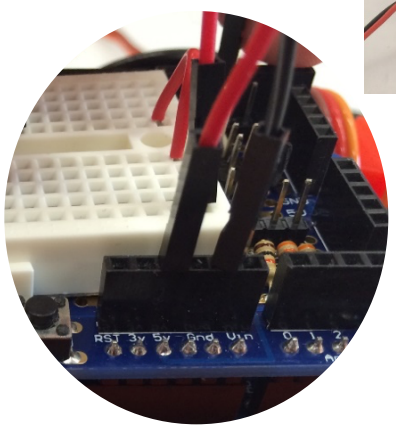


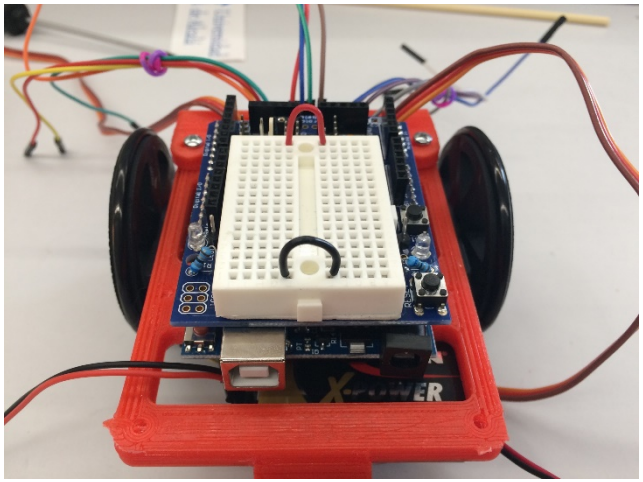
Preparación de la ProtoBoard

- Para ir montando progresivamente la electrónica, es necesario partir de unas conexiones iniciales.
  - Realiza las conexiones que se indican en la figura.



¡OJO! Los cables negros se conectan a GND (0V) y el rojo a 5V de la tarjeta



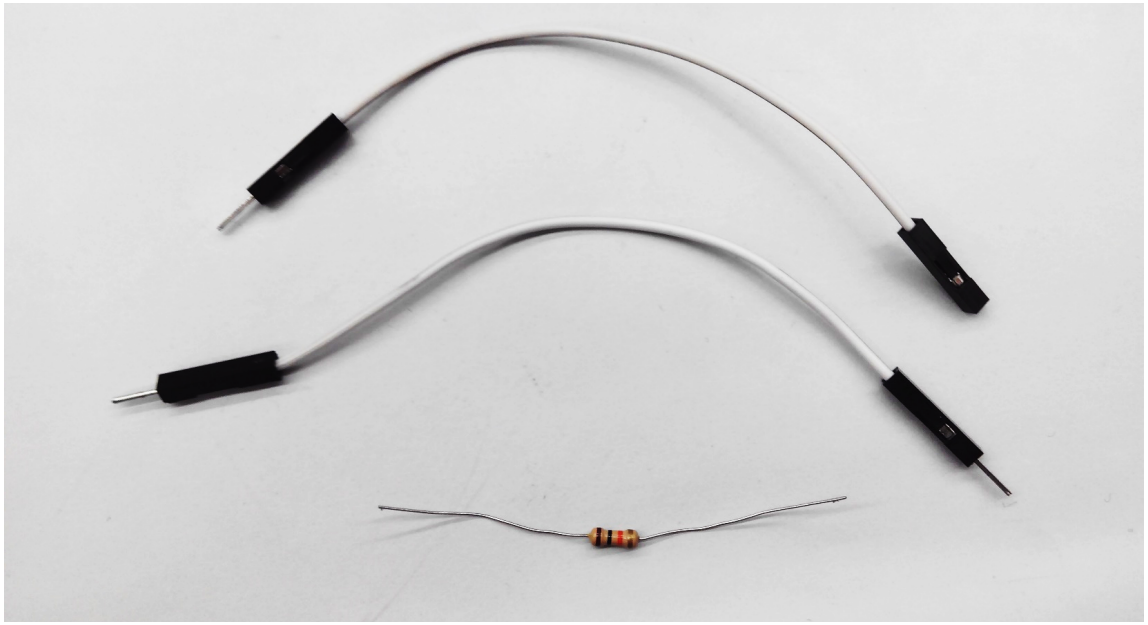


¡Asegúrate de mirar bien donde conectarlo!

Añadimos un pulsador

Añadimos un pulsador

- Materiales

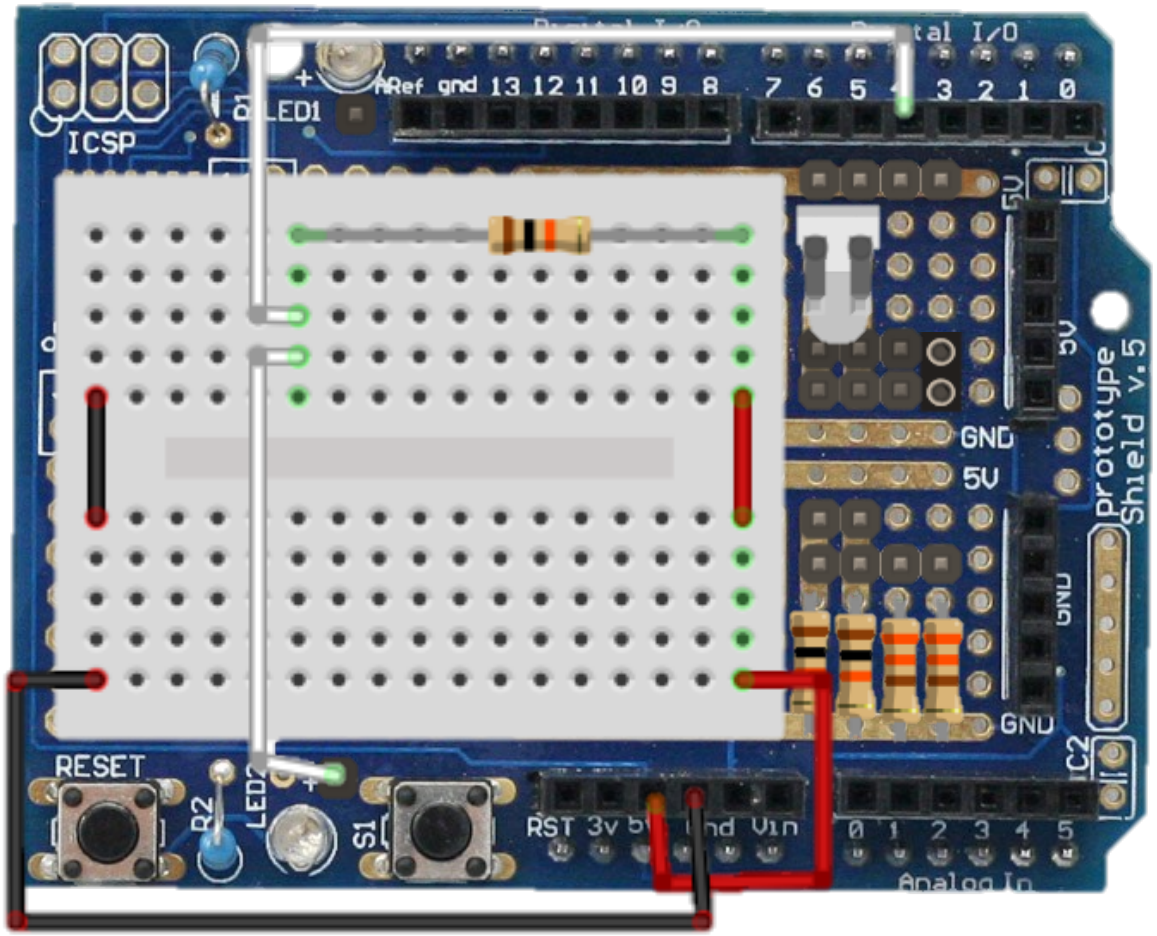


- Cable Blanco Macho-Macho (M-M)
- Cable Blanco Macho-Hembra (M-H)
- Resistencia de 10kΩ

  Marrón – **Negro** – **Naranja** - **Oro**

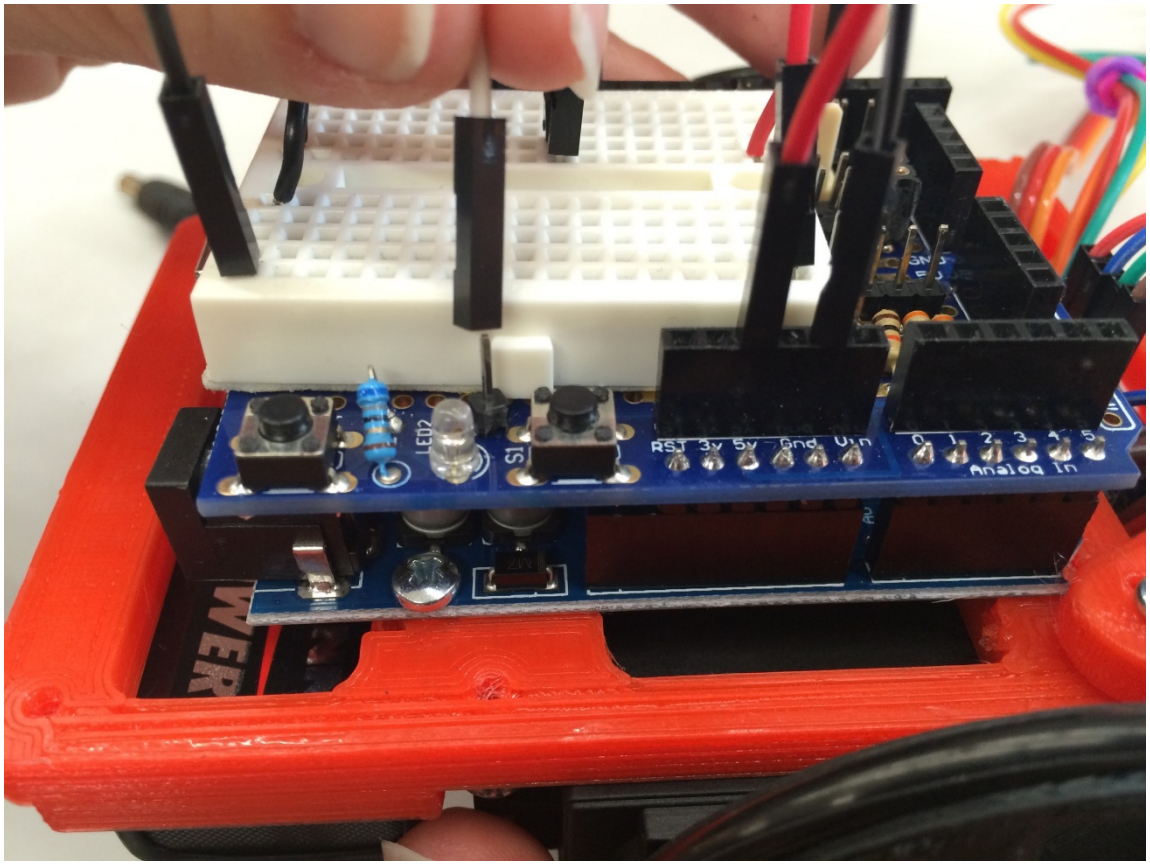
Añadimos un pulsador

- Conecta el pulsador y la resistencia como se indica en la figura. En las siguientes trasparencias se explica paso a paso.



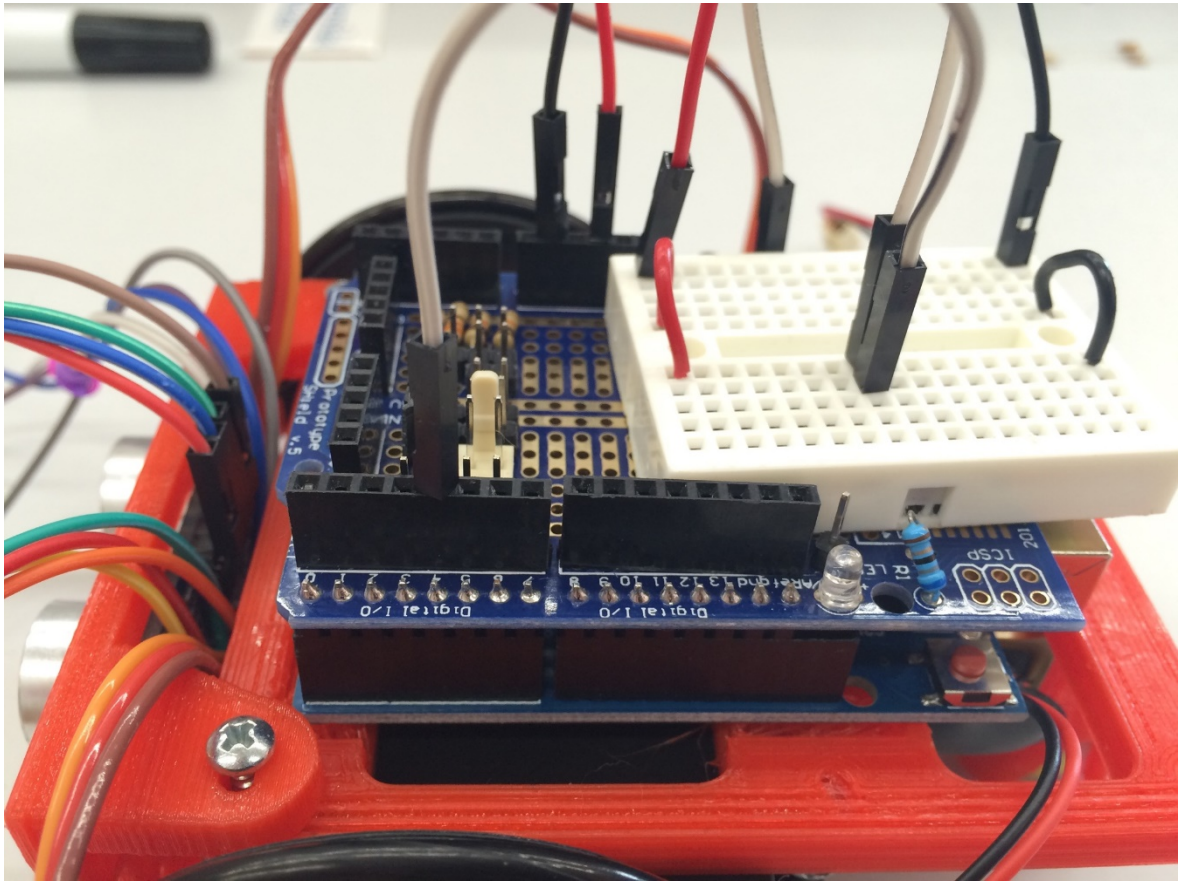
Añadimos un pulsador

- Conecta el extremo hembra del cable (M-H) al pin del pulsador. El otro extremo macho llévalo a un punto de la protoboard.



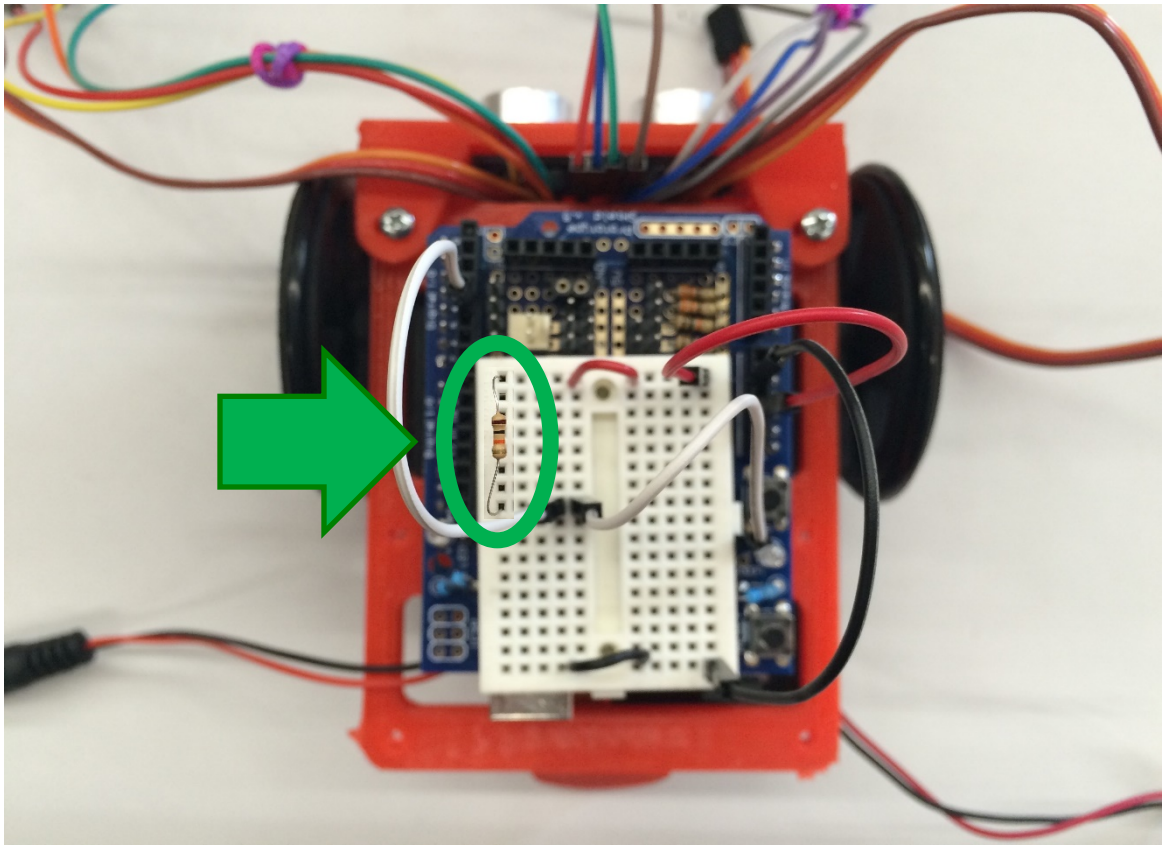
Añadimos un pulsador

- Conecta el otro cable blanco M-M entre el punto de la protoboard y el pin 4 de la placa arduino.



Añadimos un pulsador

- ¡NO TE OLVIDES DE CONECTAR LA RESISTENCIA!
  - Entre Vcc y el punto de interconexión de la protoboard irá conectada la resistencia de 10kOhm



Utilizamos el pulsador

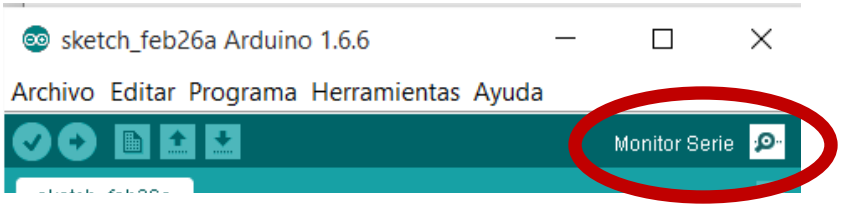
- Ejemplo: Estado de botón externo (PulsadorSerial).
  - Abre el ejemplo que se llama PulsadorSerial.
    - Archivo->Ejemplos->Tubot2016Lib->2_Pulsador->PulsadorSerial
  - Verifica y carga el programa en la tarjeta de control
  - Abre el Monitor Serie seleccionado una velocidad de 19200 baudios
    - Se explica cómo hacerlo en la siguiente transparencia
  - Comprueba que al pulsar, cambia el valor

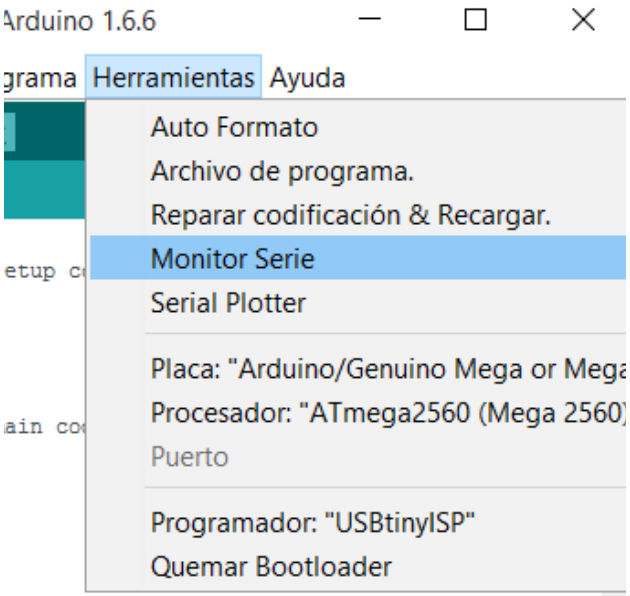
En este ejemplo leeremos el estado digital de un pulsador conectado al pin 4 de la tarjeta de control y enviaremos dicho valor el cable USB al ordenador (comunicación serie).

¿Qué valor se visualiza cuando se pulsa?

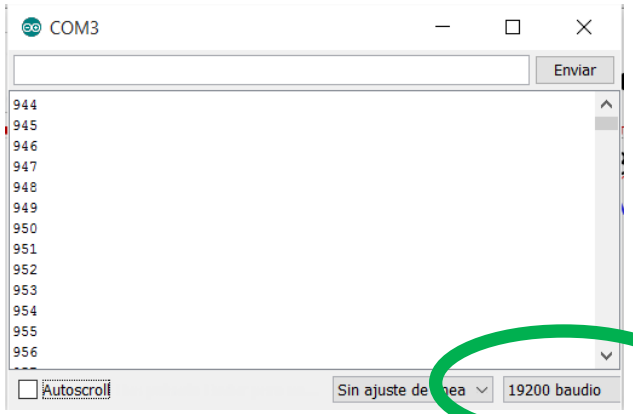
Utilizamos el pulsador

- Ejemplo: Estado de botón externo (PulsadorSerial).
  - Abre el puerto serie para conocer el valor del pulsador.
    - Herramientas → Monitor Serie
    - O bien en el botón de arriba a la izquierda.





- Debe aparecer algo así:



Selecciona la velocidad en la parte inferior derecha (19200 baudios)

Utilizamos el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorSerial).
  - El programa tiene tres partes: declaración de variables, "setup()" y "loop()"

```
/* Este programa lee el estado de un boton situado en el pin 4 y lo imprime en el Monitor Serial */

int pinPulsador = 4;  // Variable que contiene el pin donde se conecta el pulsador (entrada digital)
int pulsador;         // Variable donde se guarda el estado del pulsador
```

```
void setup(){
  //Establecemos el pin del pulsador como entrada a la tarjeta de control
  pinMode(pinPulsador, INPUT);

  //Configuración de la comunicación serie a 19200 bits por segundo (baudios).
  Serial.begin(19200);
}
```

```
void loop(){
    delay(200);

    // Leemos el valor digital del pulsador (0 ó 1) y se guarda en la variable button
    pulsador = digitalRead(pinPulsador);

    // Se envía por una comunicación serie el estado del pulsador (0 ó 1)
    // Para ver los datos hay que abrie el MonitorSerie
    Serial.print("Valor boton: ");
    Serial.println(pulsador);
}
```

Utilizamos el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorSerial).
  - Fíjate en las siguientes partes del código para que puedas entenderlas.

```
/* Este programa lee el estado de un boton situado en el pin 4 y lo imprime en el Monitor Serial */

int pinPulsador = 4;  // Variable que contiene el pin donde se conecta el pulsador (entrada digital)
int pulsador;         // Variable donde se guarda el estado del pulsador
```

```
void setup(){
  //Establecemos el pin del pulsador como entrada a la tarjeta de control
  pinMode(pinPulsador, INPUT);

  //Configuración de la comunicación serie a 19200 bits por segundo (baudios).
  Serial.begin(19200);
}
```

Configuramos el pin como entrada digital.

```
void loop(){
    delay(200);

    // Leemos el valor digital del pulsador (0 ó 1) y
    pulsador = digitalRead(pinPulsador);

    // Se envía por una comunicación serie el estado
    // Para ver los datos hay que abrie el MonitorSer
    Serial.print("Valor boton: ");
    Serial.println(pulsador);
}
```

- **digitalRead** Esta función lee el estado digital del pin dado como parámetro.
  - Devuelve el estado del pin:
    - 1 → Sí está a nivel alto ('HIGH' ó 5V)
    - 0 → Sí está a nivel bajo ('LOW' ó 0V)

Utilizamos el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorSerial).
  - El puerto serie (Serial) nos sirve para comunicarnos con el robot a través del cable USB.

```
/* Este programa lee el estado de un boton situado en el pin 4 y lo imprime en el Monitor Serial */

int pinPulsador = 4;  // Variable que contiene el pin donde se conecta el pulsador (entrada digital)
int pulsador;         // Variable donde se guarda el estado del pulsador
```

```
void setup(){
  //Establecemos el pin del pulsador como entrada a la tarjeta de control
  pinMode(pinPulsador, INPUT);

  //Configuración de la comunicación serie a 19
  Serial.begin(19200);
}
```

Serial.begin(tasa): Esta función configura la comunicación serie con una velocidad de¨**tasa**¨ bits por segundo

```
void loop(){
    delay(200);

    // Leemos el valor digital del pulsador (0 ó
    pulsador = digitalRead(pinPulsador);

    // Se envía por una comunicación serie el es
    // Para ver los datos hay que abrie el Monit
    Serial.print("Valor boton: ");
    Serial.println(pulsador);
}
```

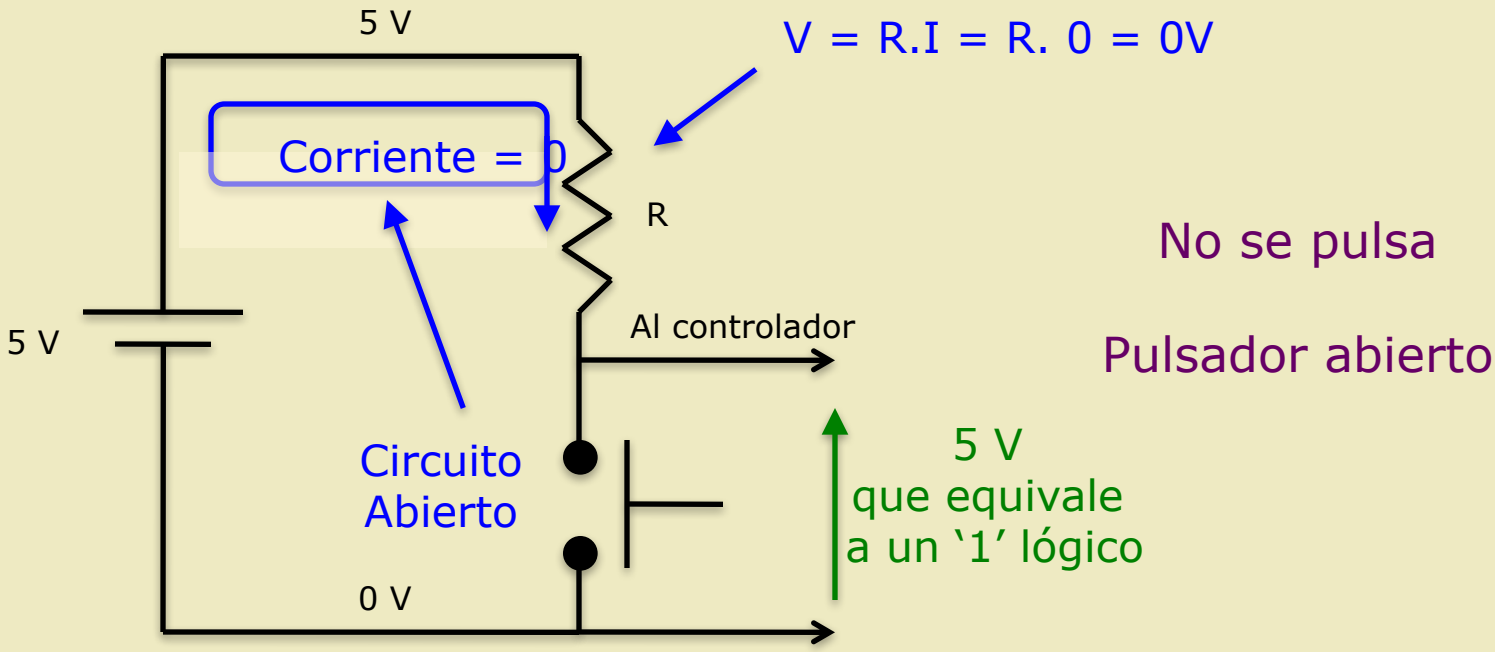
Serial.print(mensaje): Esta función envía por la comunicación serie el contenido del parámetro **"mensaje".**

- ¨mensaje¨ puede ser un texto o una variable.
  - Texto: Palabras entre comillas dobles ¨¨. Se envían dicho mensaje las palabras
  - Variable: Envía el contenido de la variable .

Serial.println(mensaje): Envía al final un salto de línea

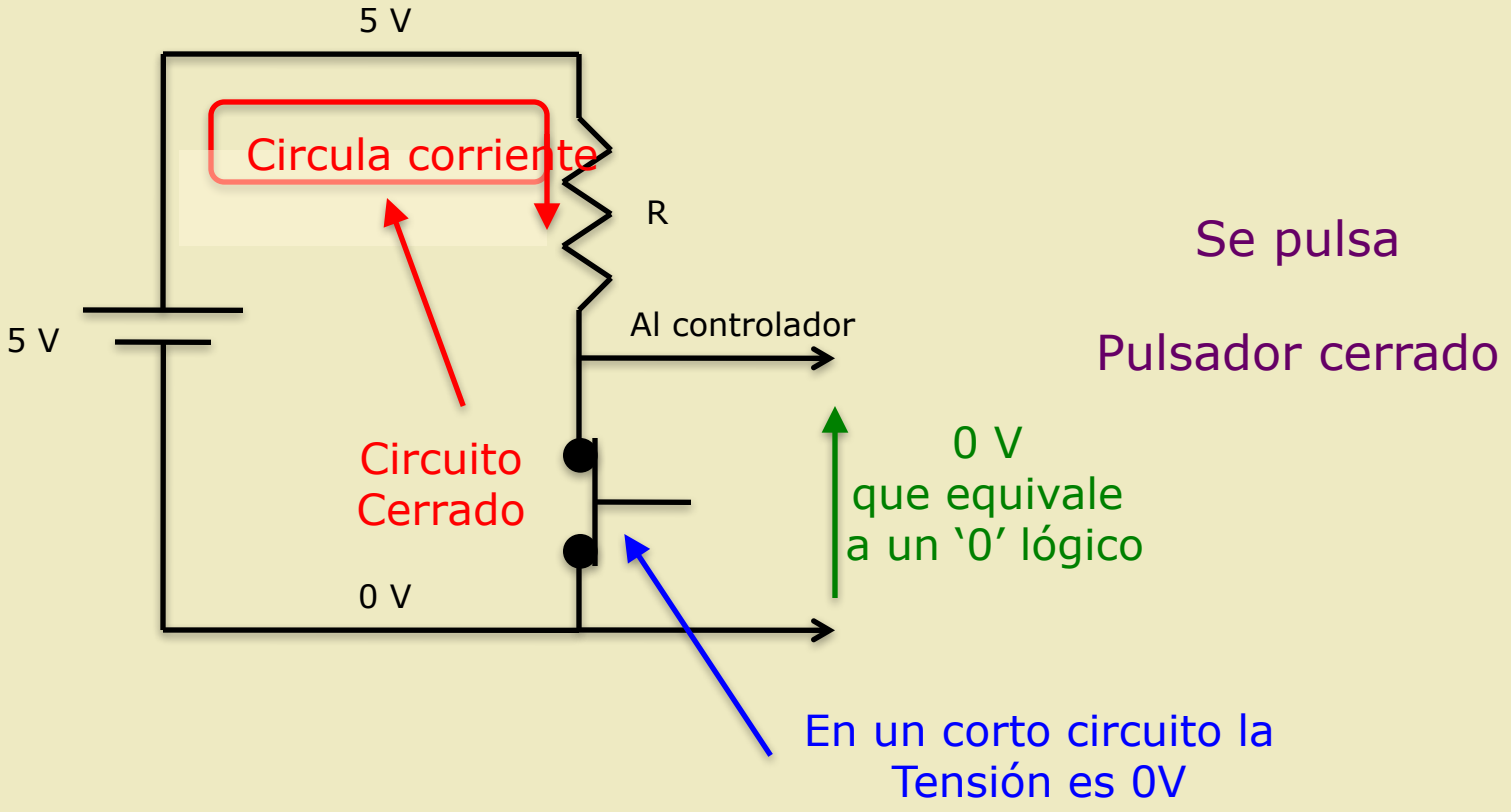
Entendiendo al pulsador

- Circuito eléctrico del pulsador
  - Cuando se pulsa se cierra el circuito cuando no está abierto
  - Se conecta a una entrada digital
  - La Ley de Ohm: V = R . I



Entendiendo al pulsador

- Circuito eléctrico del pulsador
  - Cuando se pulsa se cierra el circuito cuando no está abierto
  - Se conecta a una entrada digital
  - La Ley de Ohm: V = R . I



Seguimos utilizando el pulsador

- Ejemplo: Control del LED con el pulsador (PulsadorLED).
  - Abre el ejemplo del botón que se llama PulsadorLED.
    - Archivo->Ejemplos->Tubot2016Lib->2_Pulsador->PulsadorLED
  - Verifica y carga el programa en la tarjeta de control
  - Comprueba que al pulsar se enciende el LED de la tarjeta de control (el de abajo).

Seguimos utilizando el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorLED).
  - De nuevo el programa tiene tres partes

```
int pinPulsador = 4;  // El pulsador se conecta al pin 4
int pinLed = 13;      // El led se conecta al pin 13
int pulsador;         // Variable donde se guarda el estado del botón
```

```
void setup(){

  // Se configura el pin donde se conecta el pulsador como entrada
  pinMode(pinPulsador, INPUT);

  // Se configura el pin donde se conecta el LED como salida
  pinMode(pinLed, OUTPUT);
}
```

```
void loop(){
    delay(200);

    // Leemos el valor digital del pulsador (0 o 1).
    pulsador = digitalRead(pinPulsador);

    if (pulsador==0)  //si el boton no esta pulsado
    {
       digitalWrite(pinLed, HIGH); //Encendemos el LED
    }
    else //si el boton esta pulsado
    {
       digitalWrite(pinLed, LOW); //Apagamos el LED
    }
}
```

Seguimos utilizando el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorLED).
  - De nuevo el programa tiene tres partes

```
int pinPulsador = 4;  // El pulsador se conecta al pin 4
int pinLed = 13;      // El led se conecta al pin 13
int pulsador;         // Variable donde se guarda el estado del botón

void setup(){

  // Se configura el pin donde se conecta el pulsador como entrada
  pinMode(pinPulsador, INPUT);

  // Se configura el pin donde se conecta el LED como salida
  pinMode(pinLed, OUTPUT);
}

void loop(){
    delay(200);

    // Leemos el valor digital del pulsador (0 o 1).
    pulsador = digitalRead(pinPulsador);

    if (pulsador==0)  //si el boton no esta pulsado
    {
       digitalWrite(pinLed, HIGH); //Encendem
    }
    else //si el boton esta pulsado
    {
       digitalWrite(pinLed, LOW); //Apagamos
    }
}
```

Configuramos el pin del pulsador como entrada digital.

Configuramos el pin del LED como salida digital.

- **digitalRead** Guarda el estado del pulsador en la variable
  - 0 → pulsado
  - 1 → no pulsado

Enciende el LED poniendo 5V en la salida digital

Apaga el LED poniendo 0V en la salida digital

Seguimos utilizando el pulsador

- Ejemplo: Estado de pulsador externo (PulsadorLED).
  - De nuevo el programa tiene tres partes

```
int pinPulsador = 4;  // El pulsador se conecta al pin 4
int pinLed = 13;      // El led se conecta al pin 13
int pulsador;         // Variable donde se guarda el estado del botón
```

```
void setup(){

  // Se configura el pin donde se conecta el pulsador como entrada
  pinMode(pinPulsador, INPUT);

  // Se configura el pin donde se conecta el LED como salida
  pinMode(pinLed, OUTPUT);
}
```

```
void loop(){
    delay(200);

    // Leemos el valor digital del pulsado
    pulsador = digitalRead(pinPulsador);

    if (pulsador==0)  //Si el boton no est
    {
       digitalWrite(pinLed, HIGH); //Encen
    }
    else //si el boton esta pulsado
    {
       digitalWrite(pinLed, LOW); //Apagamos el LED
    }
}
```

- Esta estructura permite programar condiciones
- **If (condición)**
  { Se ejecuta esto si se cumple la condición
  }
  else
  { Se ejecuta esto si NO se cumple la condición
  }

La condición de igualdad se pone como ==

Puedes encontrar más información sobre la sentencia if en
https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C/Instrucciones_de_control

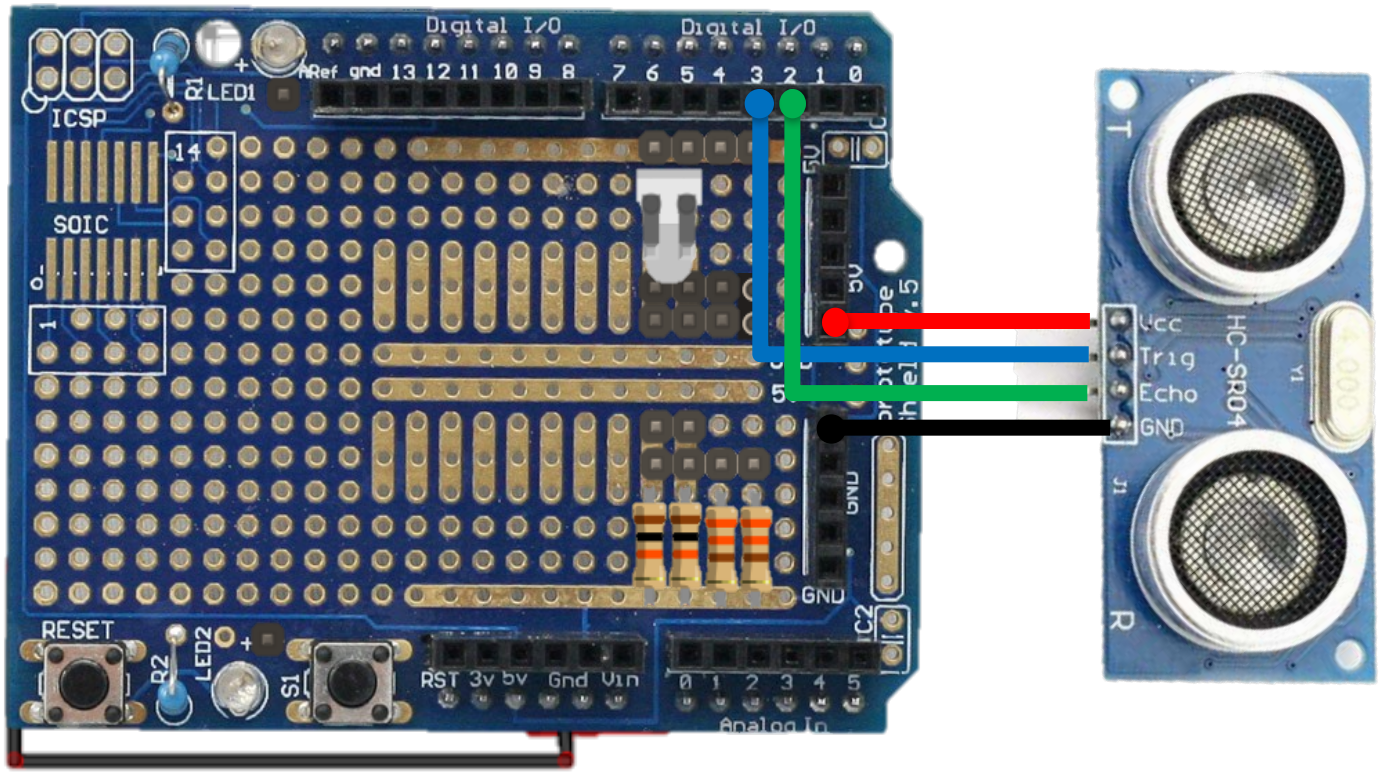
Seguimos utilizando el pulsador

- **Ejercicios**:
  1. Modifica el programa PulsadorLED para que al pulsar se encienda el LED de la tarjeta de conexiones.
     - Escribe, verifica el programa y cárgalo en la placa
  2. Modifica el programa para que al pulsar se encienda un LED y al soltar el otro LED.
     - Escribe, verifica el programa y cárgalo en la placa
  3. Realiza un programa que haga parpadear un LED a una frecuencia y que al pulsar vaya el doble de rápido.
     - Escribe, verifica el programa y cárgalo en la placa
  4. Realiza un programa para que con una pulsación corta, encienda un LED si estaba apagado y lo apague si estaba encendido.
     - Escribe, verifica el programa y cárgalo en la placa

Conectamos el sensor de Ultrasonidos

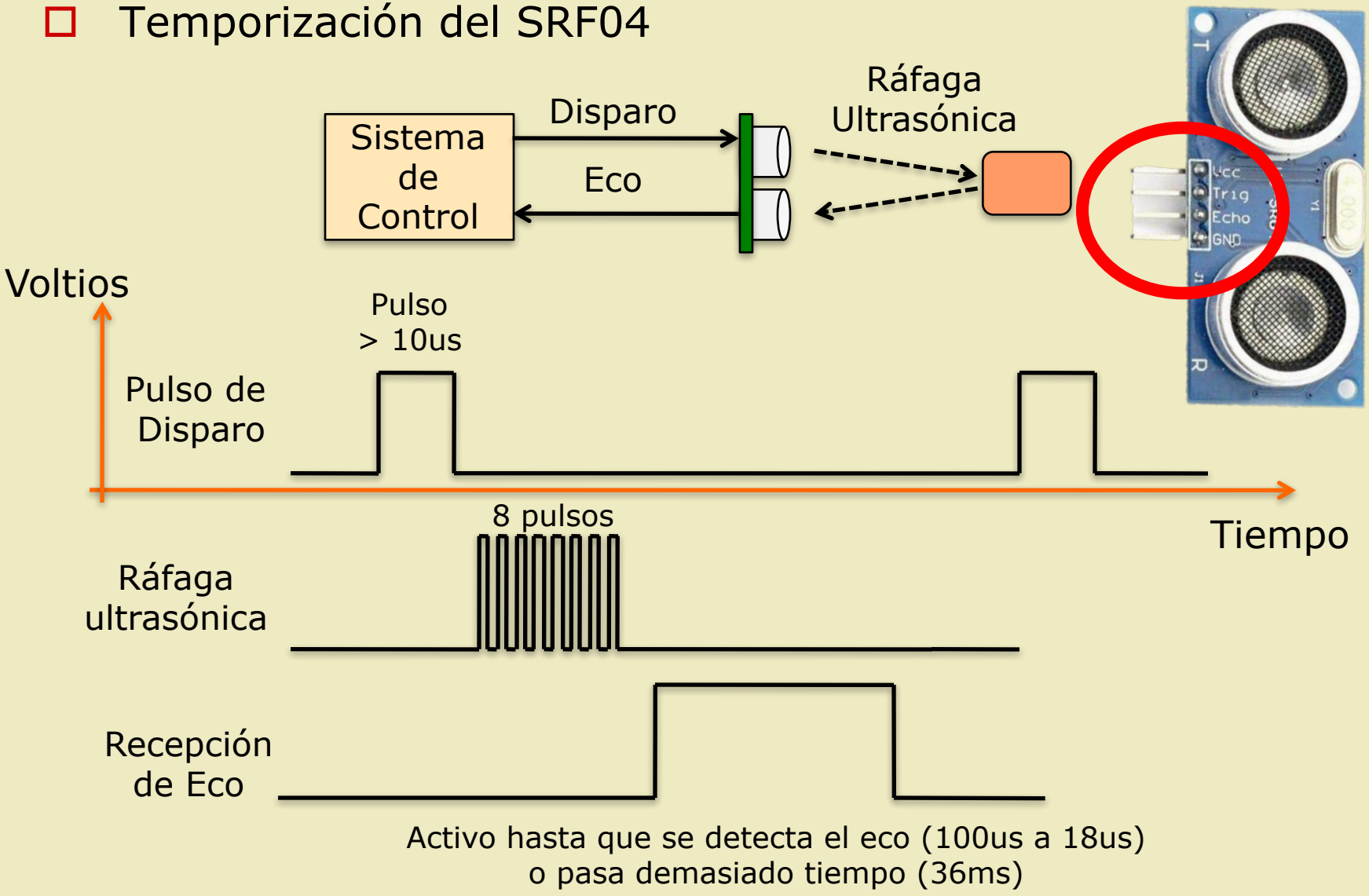
Jugamos con el sensor de Ultrasonidos

- Conecta el sensor de ultrasonidos como se indica en la figura.
  - Vcc del sensor va a 5V, Trig va al Pin 3, Echo al Pin 2 y GND a GND



Entendiendo el sensor de Ultrasonidos

- Temporización del SRF04



Activo hasta que se detecta el eco (100us a 18us)
o pasa demasiado tiempo (36ms)

Jugamos con el sensor de Ultrasonidos

- Ejemplo: Sensor de Distancia (DistanciaSerial)
  - En este ejemplo leeremos la distancia medida por el SRF04

```
#include <DistanceSRF04.h> // Incluímos librería del SRF04
// Pines del SRF04
int pinEco  =           2;
int pinTrigger =        3;
// Variables para el control del sensor
DistanceSRF04 Dist;
int distance;

void setup(){
  Serial.begin(19200);
  // Configuramos los pines del sensor.
  Dist.begin(pinEco,pinTrigger);
}

void loop(){
  delay(200);
  // Guardamos la distancia del sensor en la variable
  distance = Dist.getDistanceCentimeter();
  // Enviamos el valor de la distancia por la comunicación serie
  Serial.print("Distancia  ");
  Serial.print(distance);
  Serial.println("cm");
}
```

Jugamos con el sensor de Ultrasonidos

- Ejemplo: Sensor de Distancia (DistanciaSerial)
  - En este ejemplo leeremos la distancia medida por el SRF04

```
#include <DistanceSRF04.h> // Incluímos librería del SRF04
// Pines del SRF04
int pinEco  =           2;
int pinTrigger =        3;
// Variables para el control del sensor
DistanceSRF04 Dist;
int distance;

void setup(){
  Serial.begin(19200);
  // Configuramos los pines del sensor.
  Dist.begin(pinEco,pinTrigger);
}

void loop(){
  delay(200);
  // Guardamos la distancia del sensor en la variable
  distance = Dist.getDistanceCentimeter();
  // Enviamos el valor de la distancia por la comunicación serie
  Serial.print("Distancia  ");
  Serial.print(distance);
  Serial.println("cm");
}
```

Variable para utilizar el sensor

Variable donde guardamos la distancia medida por el sensor

Configuramos el sensor SRF04 conectándolo a los pines

Obtenemos la distancia medida por el sensor

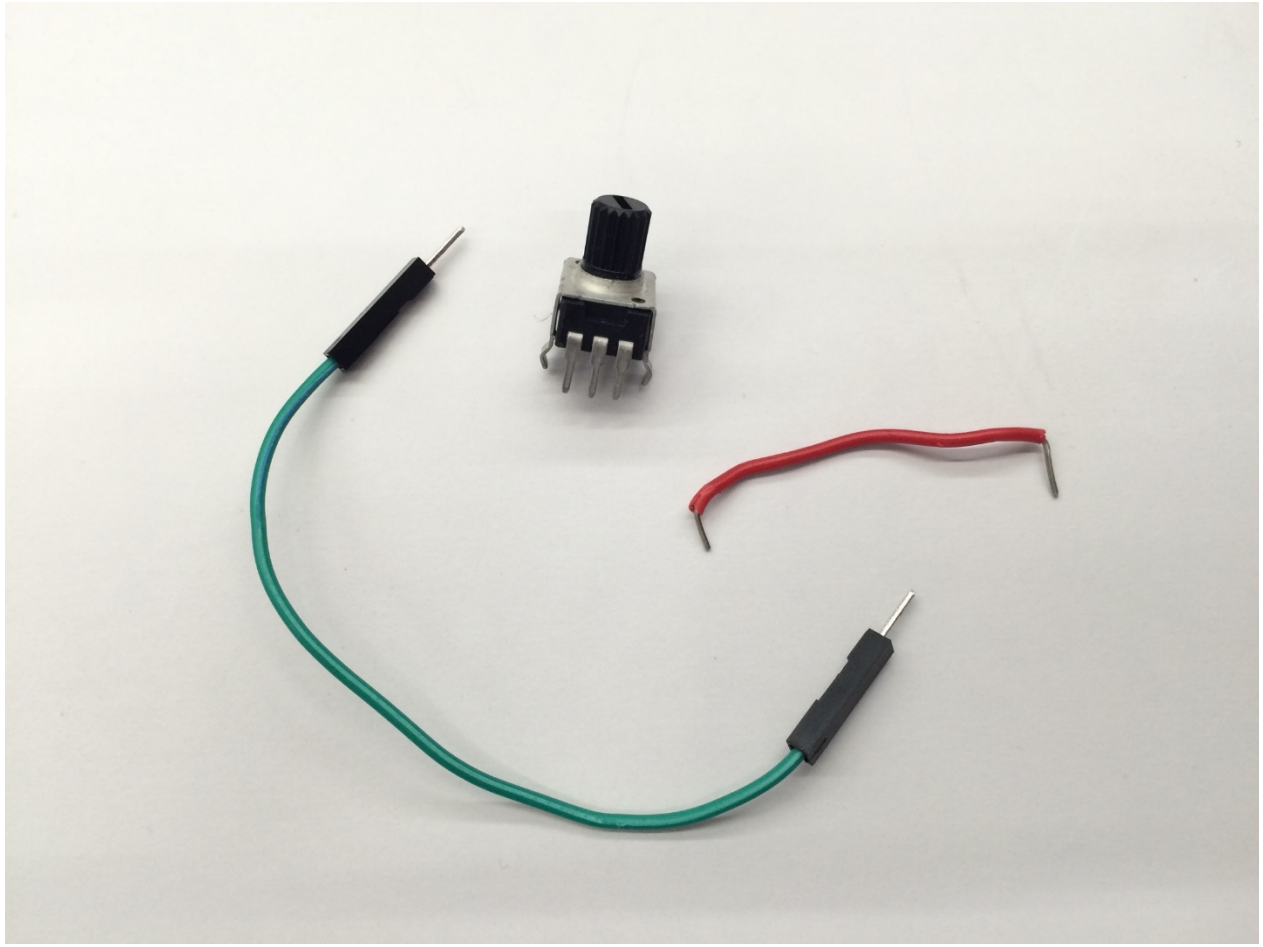
Seguimos jugando con el sensor de ultrasonidos

- **Ejercicios**:
  1. Modifica el programa DistanciaSerial para cuando se detecte un objeto más cerca de 5cm se encienda un LED
     - Escribe, verifica el programa y cárgalo en la placa

El potenciómetro

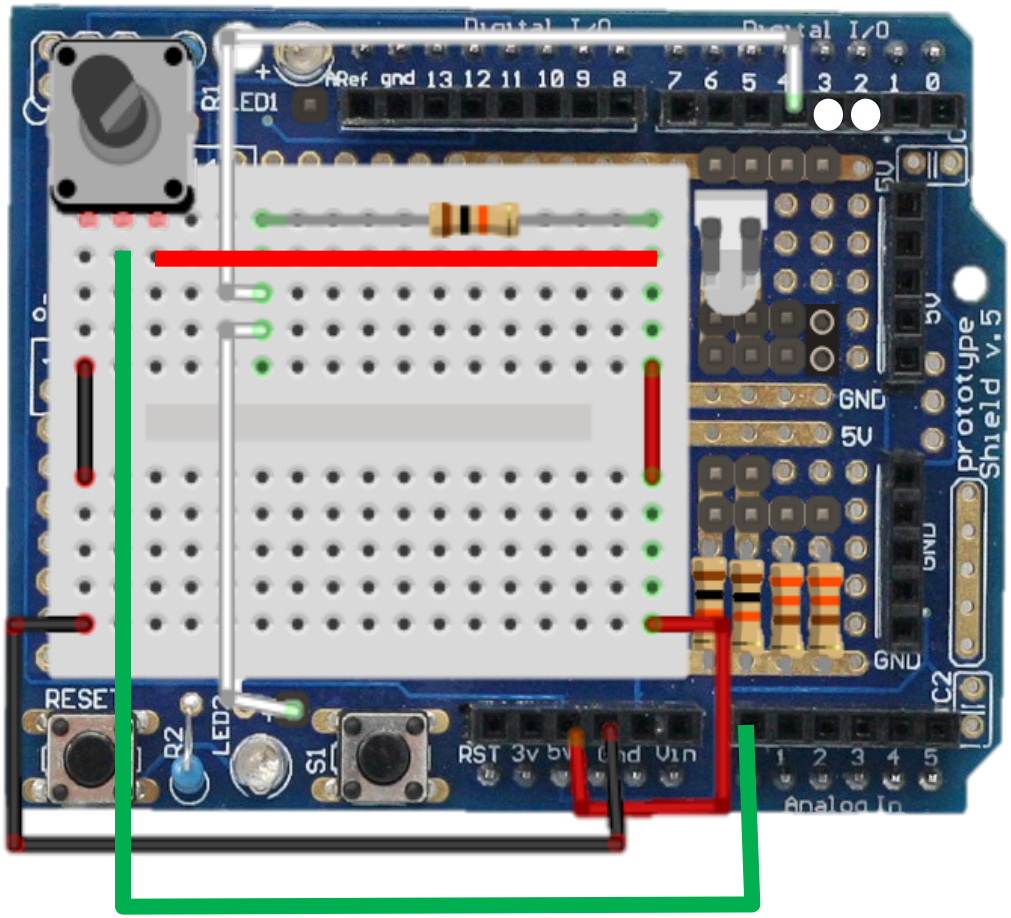
Conexión del potenciómetro

- Materiales

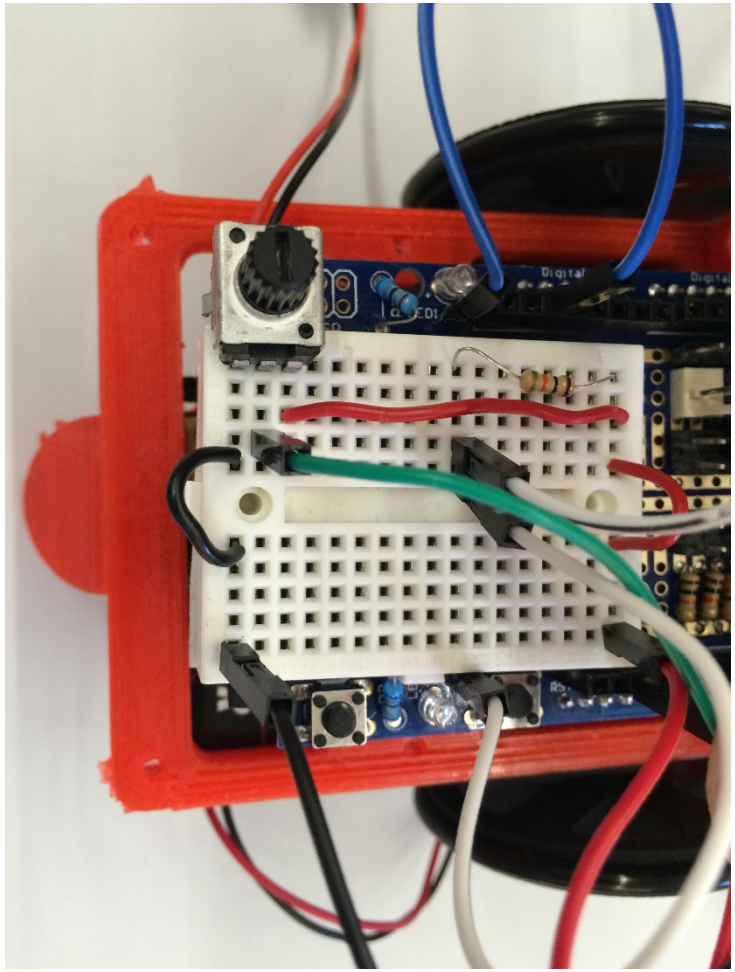


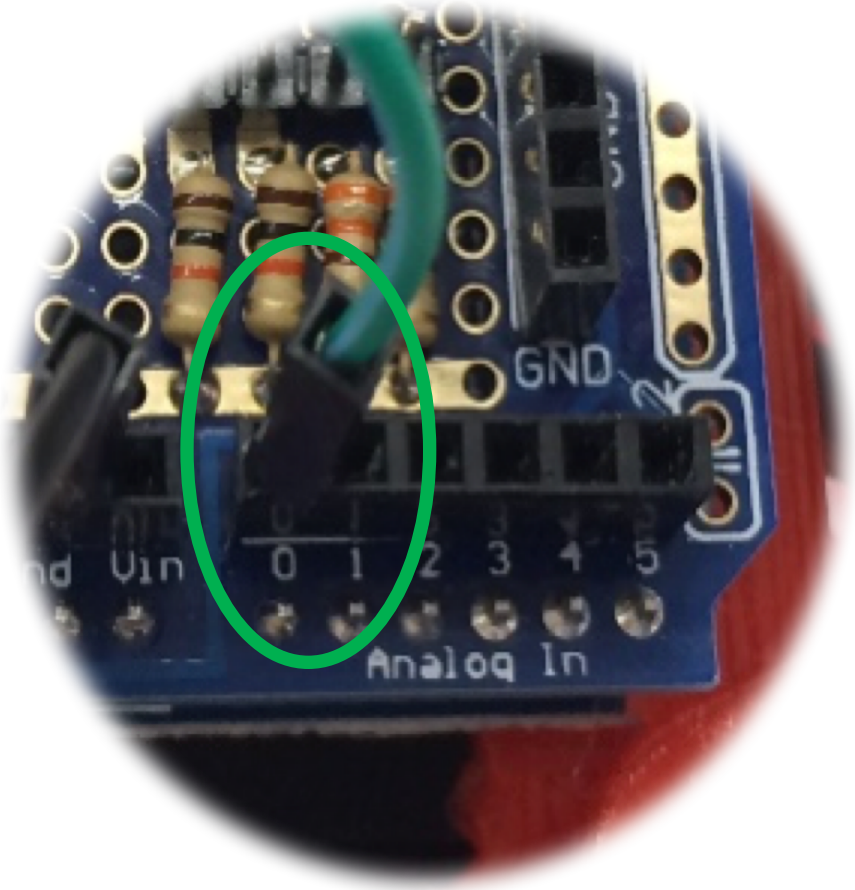
Conexión del potenciómetro

- Conecta el potenciómetro como se indica en la figura.



Conexión del potenciómetro





- Ejemplo: Potenciómetro
  - En este ejemplo leeremos la tensión que cae en el potenciómetro.

```
int pinPotenciometer = A0;
int potenciometer;

void setup(){
  Serial.begin(19200); // Configuramos comunicacion serie
}

void loop()
{
  float calculo;
  potenciometer = analogRead(pinPotenciometer);
  calculo = (float)potenciometer*(5.0/1023);
  Serial.println("Potenciometro ");
  Serial.print(calculo);
  Serial.println("\n");
  delay(100);
}
```

Esta función devuelve un valor entre 0 y 1023.

Cuando la entrada tiene 0V -> 0
Cuando tiene 5V -> 1023

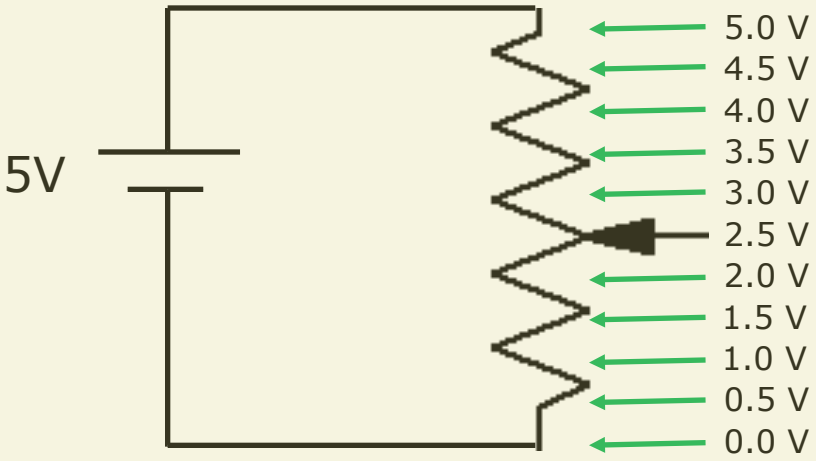
Esta función realiza la conversión a voltios.
La regla de tres es la siguiente:

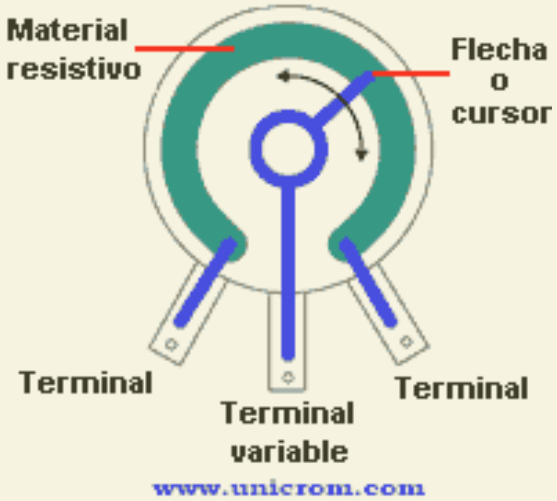
1023 --------- 5V
Potenciometer ------- Calculo

Calculo=potenciometer*(5/1023)

Entendiendo el potenciómetro

- ¿Por qué al mover el potenciómetro cambia el valor que leemos?
  - Cuando se pone una tensión (voltios) en los extremos de una resistencia, la tensión va cambiado según se avanza en la resistencia.
  - El potenciómetro permite ver la tensión en el interior de una resistencia





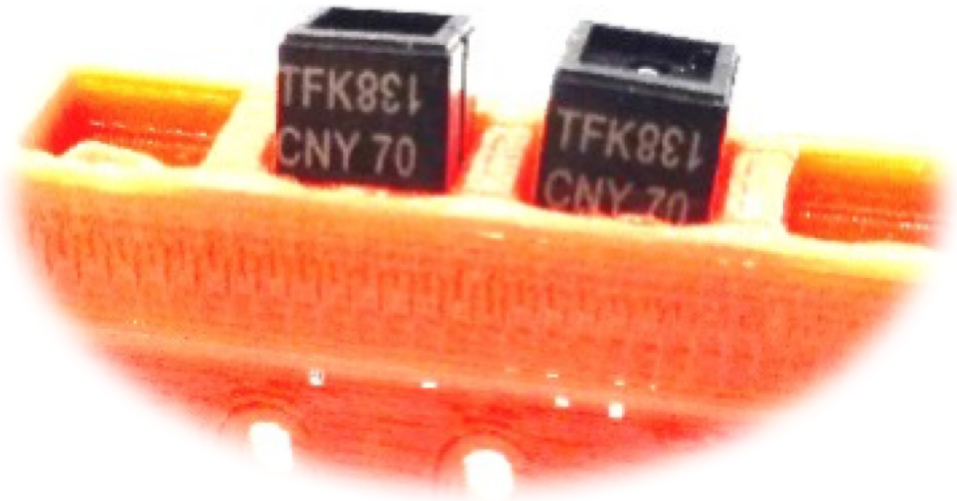
Se puede encontrar más información sobre cómo funciona un potenciómetro en …

http://unicrom.com/wp-content/uploads/descripcion_potenciometro1.gif

Seguimos jugando con potenciómetro

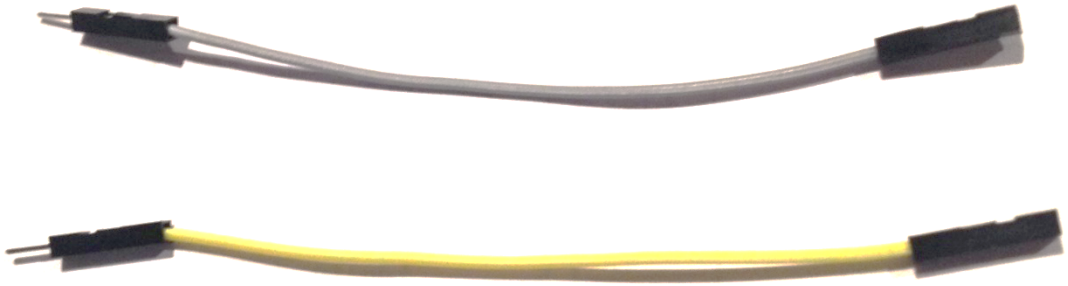
- **Ejercicios**:
  1. Modifica el programa PotenciometroSerial para que durante la mitad del recorrido esté un LED encendido y durante la otra mitad que el que esté encendido sea el otro LED
     - Escribe, verifica el programa y cárgalo en la placa
  2. Modifica el programa ParpadeaLED para que la velocidad de parpadeo se ajuste con el potenciómetro. Envía tiempo entre parpadeos por el Puesto Serie.
     - Escribe, verifica el programa y cárgalo en la placa
  3. Realiza un programa que encienda un LED cuando se detecta un objeto más cerca de una determinada posición. El valor de esta posición de referencia debe ser variable con el valor del pulsador. Envía por el Puerto Serie el valor de la medida y de la referencia.
     - Escribe, verifica el programa y cárgalo en la placa

Sensores de infrarrojos CNY70



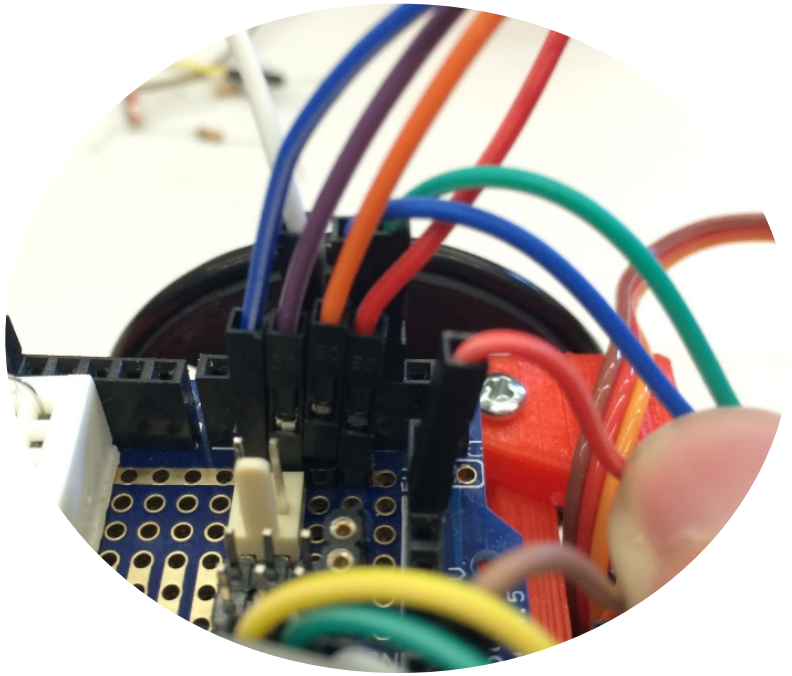
Conexión de los CNY70

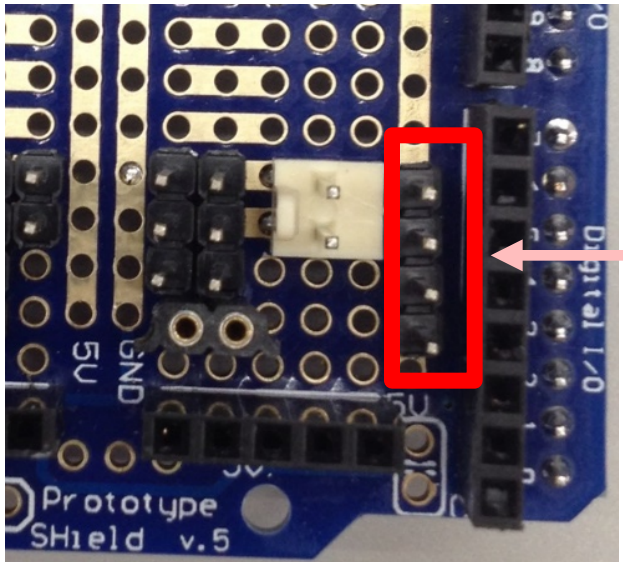
- Materiales



Conexión de los CNY70

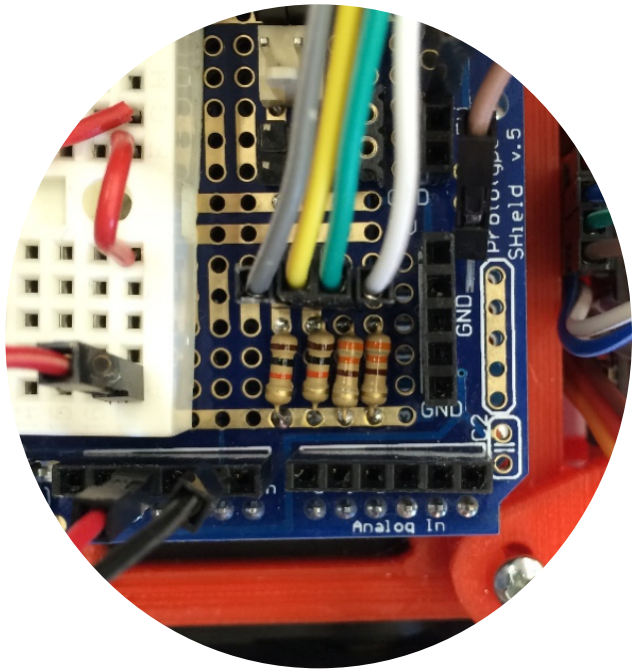
- Conectamos los cables que suben de los sensores CNY70 como se indica en la figura.
  - Los cables de alimentación (azul, morado, naranja y rojo) van conectados al conector de 5V de la tarjeta de conexiones.

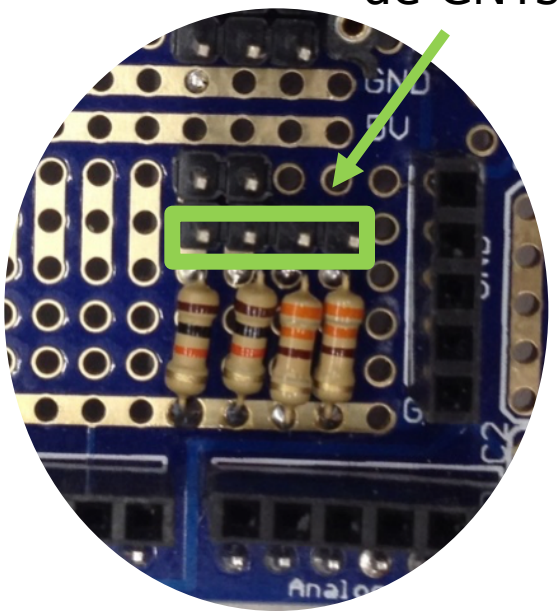




Conexión de los CNY70

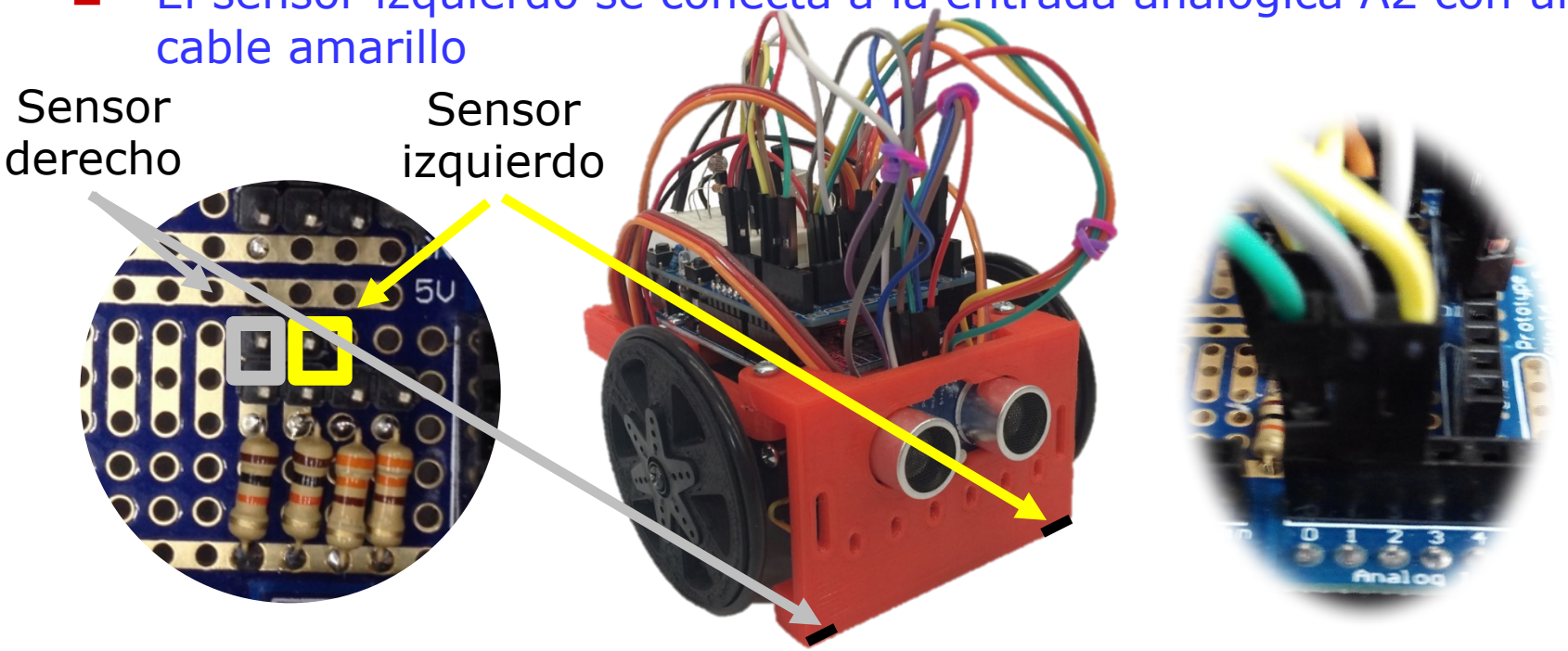
- Conectamos los cables que suben de los sensores CNY70 como se indica en la figura.
  - El resto de los cables van a las resistencias y se conectan como se indica en la figura.





Conexión de los CNY70

- Conectamos los sensores a la tarjetas de control.
  - En la figura de la izquierda puede verse el punto donde se conectan los sensores a las tarjeta de control.
  - El sensor derecho se conecta a la entrada analógica A1 con un cable gris
  - El sensor izquierdo se conecta a la entrada analógica A2 con un cable amarillo

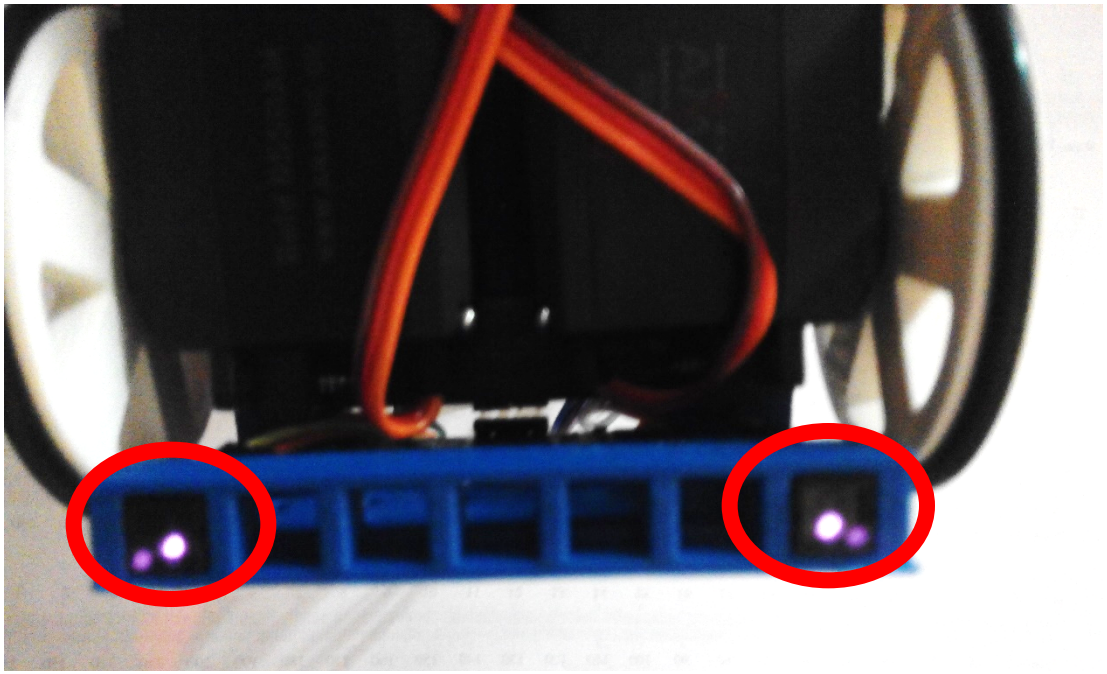


Leyendo los sensores de infrarrojos

- Ejemplo: Lectura del valor de los sensores de infrarrojos (InfraredSerial).
  - Abre el ejemplo del botón que se llama InfraredSerial.
    - Archivo->Ejemplos->Tubot2016Lib->5_Infrared->InfraredSerial
  - Verifica y carga el programa en la tarjeta de control
  - Abre el Monitor Serie configurado a 19200 baudios.
  - Comprueba que al colocar el robot con los sensores sobre una superficie blanca o sobre una superficie negra cambia el valor recibido.
    - Apunta el valor que da con los dos sensores en blanco
    - Apunta el valor que da con los dos sensores en negro

Conexión de los CNY70

- ¿Funciona?
  - Para ver si funciona vamos a usar la cámara del móvil. Nuestro ojo no puede ver los infrarrojos, pero si la cámara no tiene un filtro se puede ver como en la foto.



Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

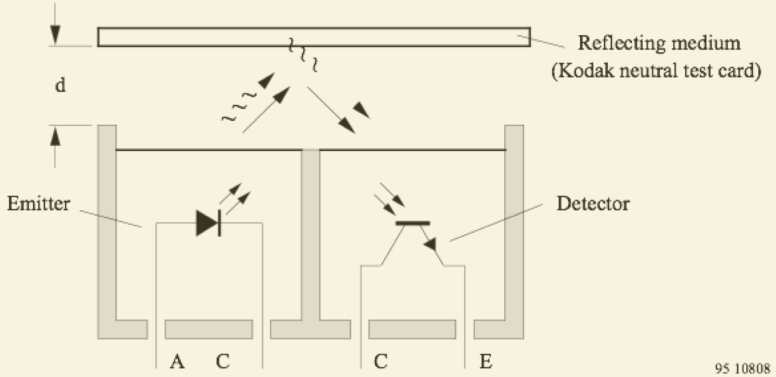
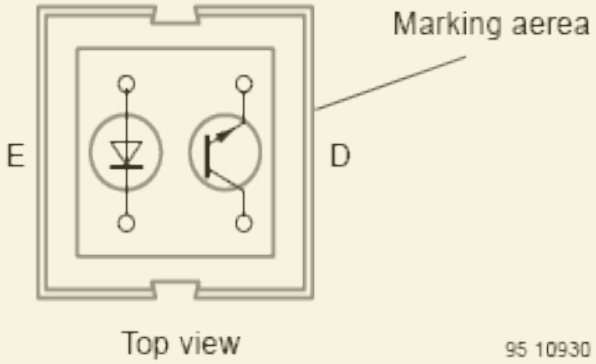
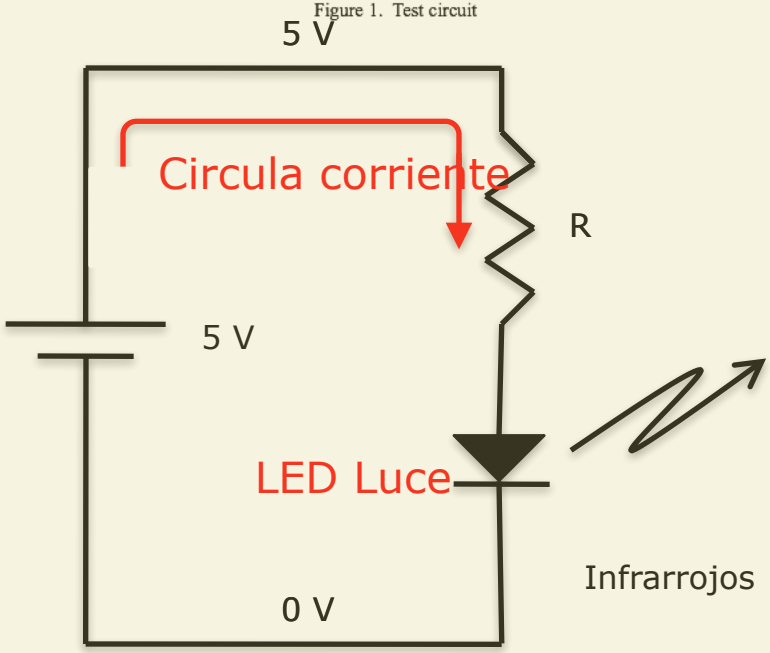


Figure 1. Test circuit





Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

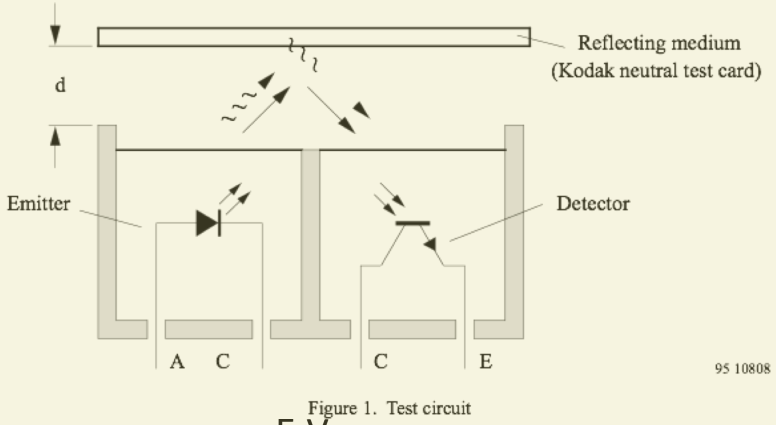
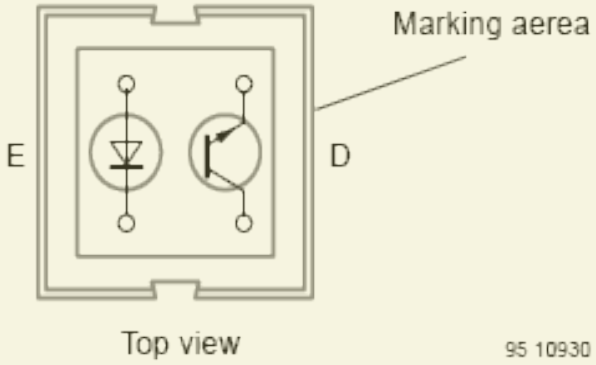
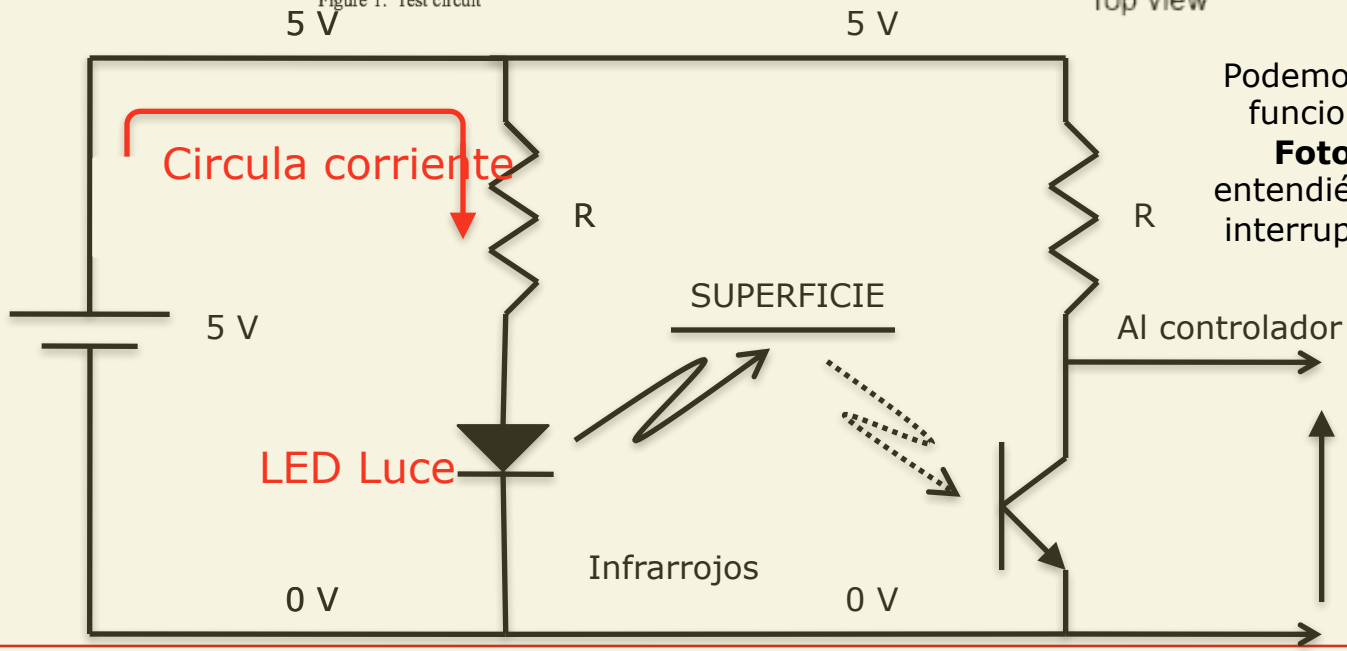


Figure 1. Test circuit





Podemos simplificar el funcionamiento del **Fototransistor** entendiéndolo como un interruptor controlado por luz

Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

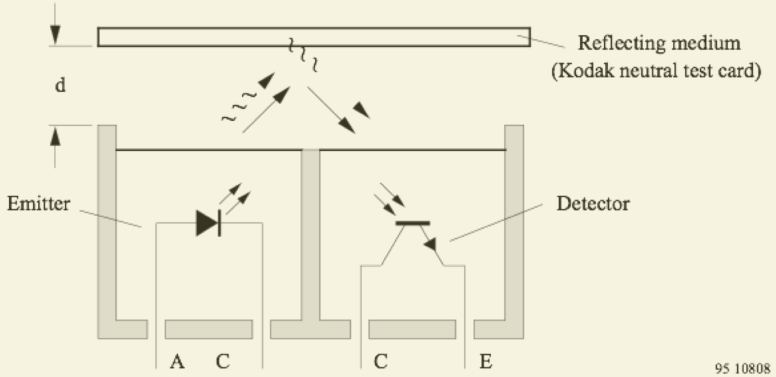
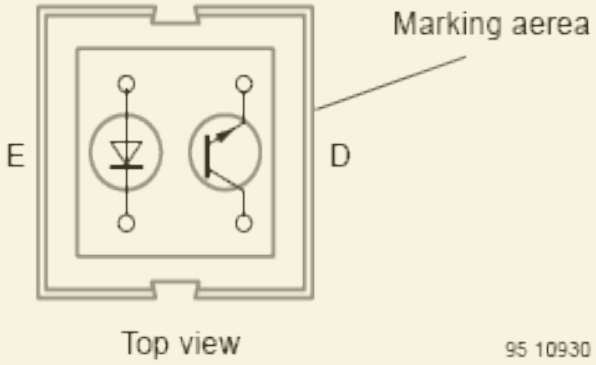
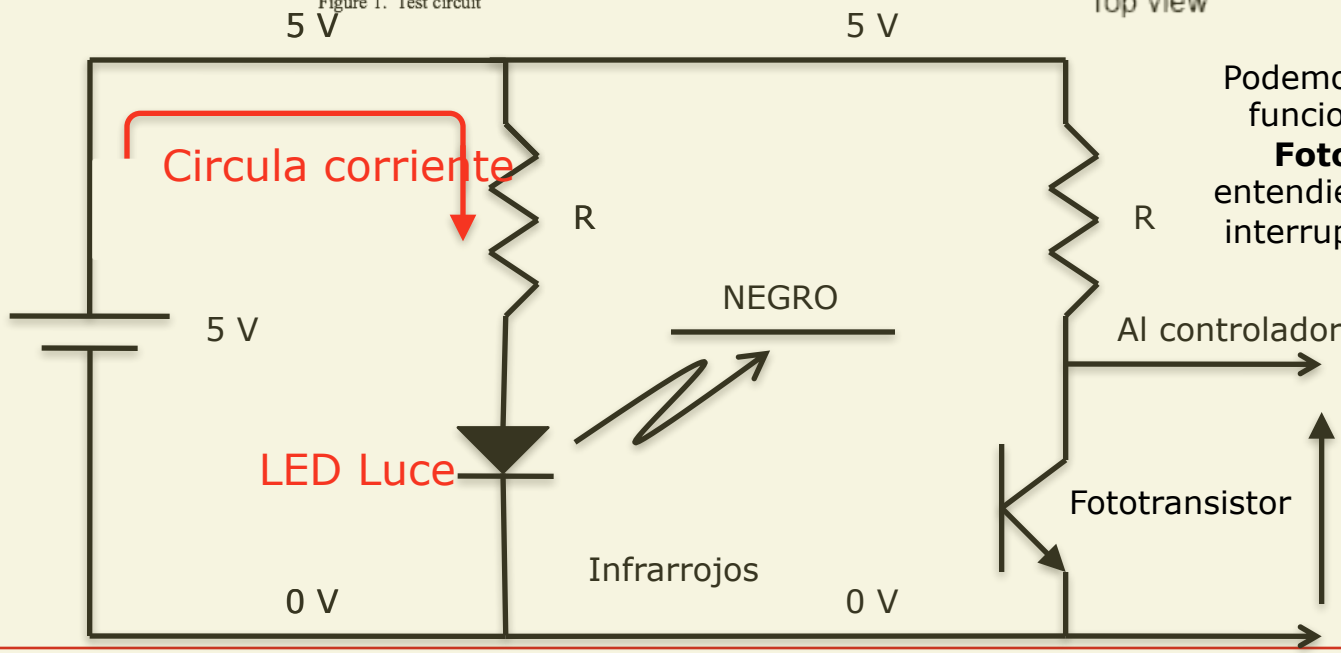


Figure 1. Test circuit





Podemos simplificar el funcionamiento del **Fototransistor** entendiéndolo como un interruptor controlado por luz

Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

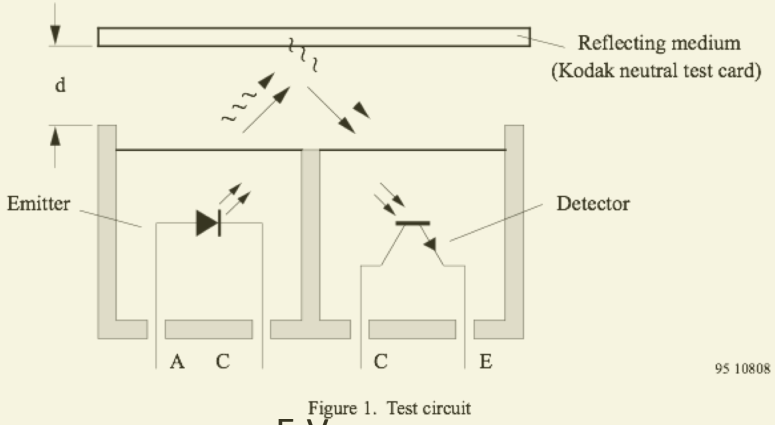
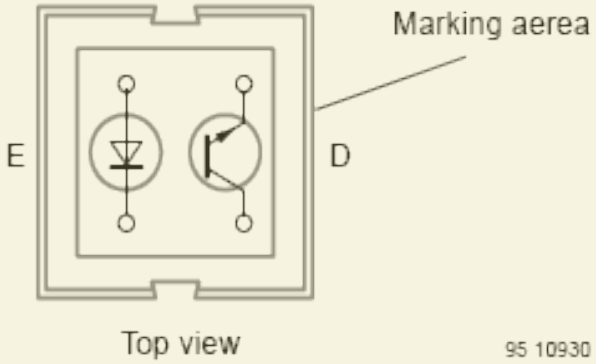
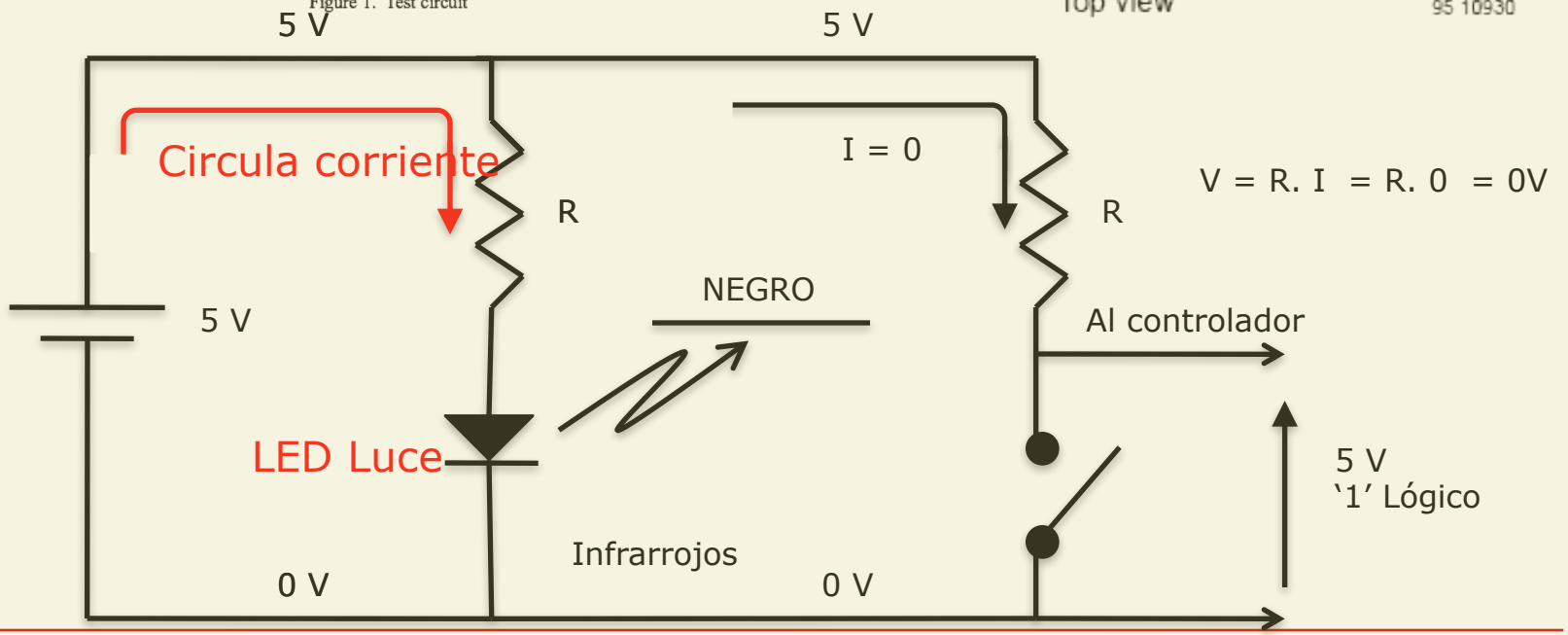


Figure 1. Test circuit





Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

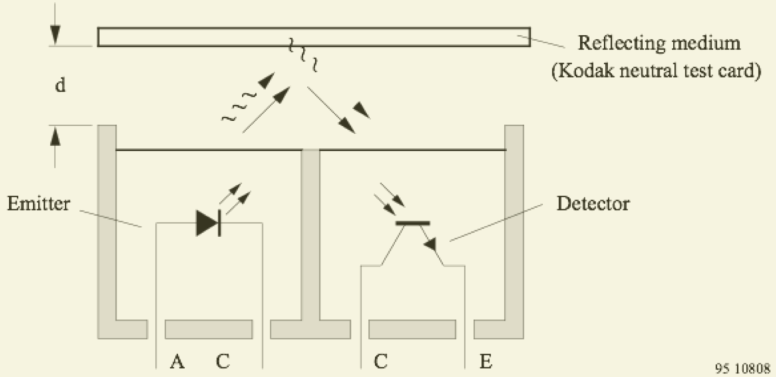
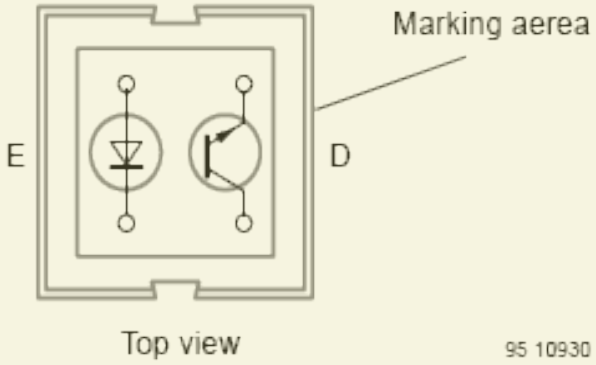
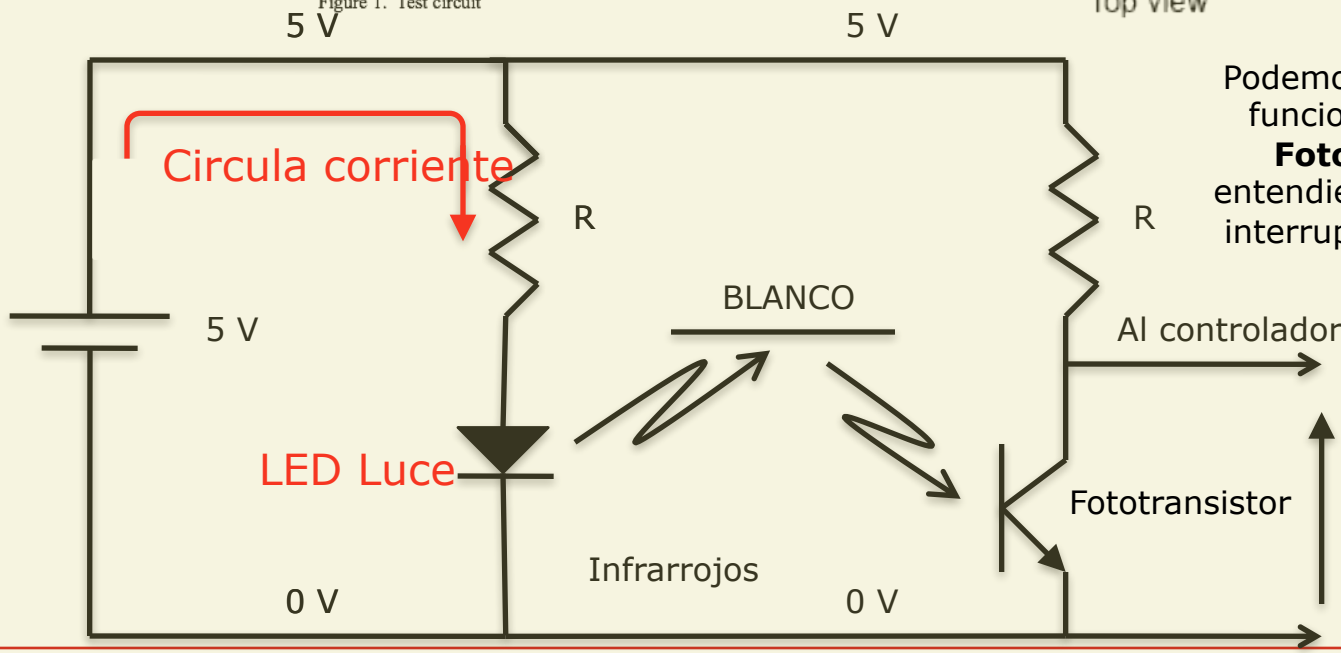


Figure 1. Test circuit





Podemos simplificar el funcionamiento del **Fototransistor** entendiéndolo como un interruptor controlado por luz

Entendiendo los sensores CNY70

- Sensor de infrarrojos reflectivo: detección de blanco o negro

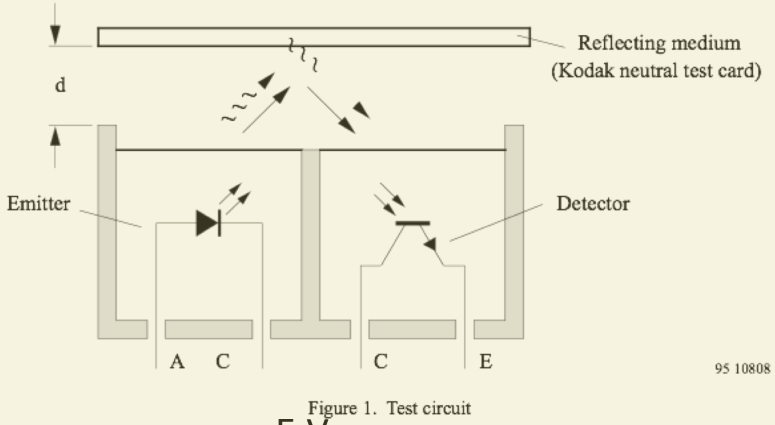
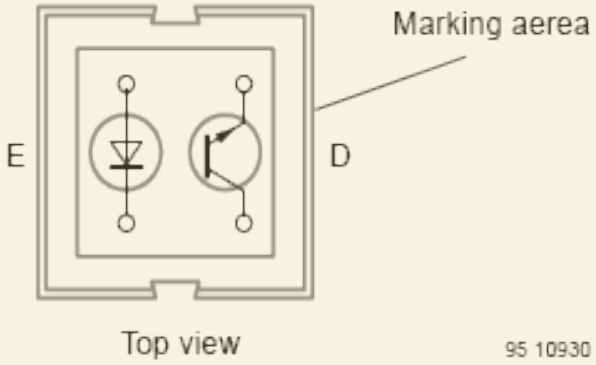
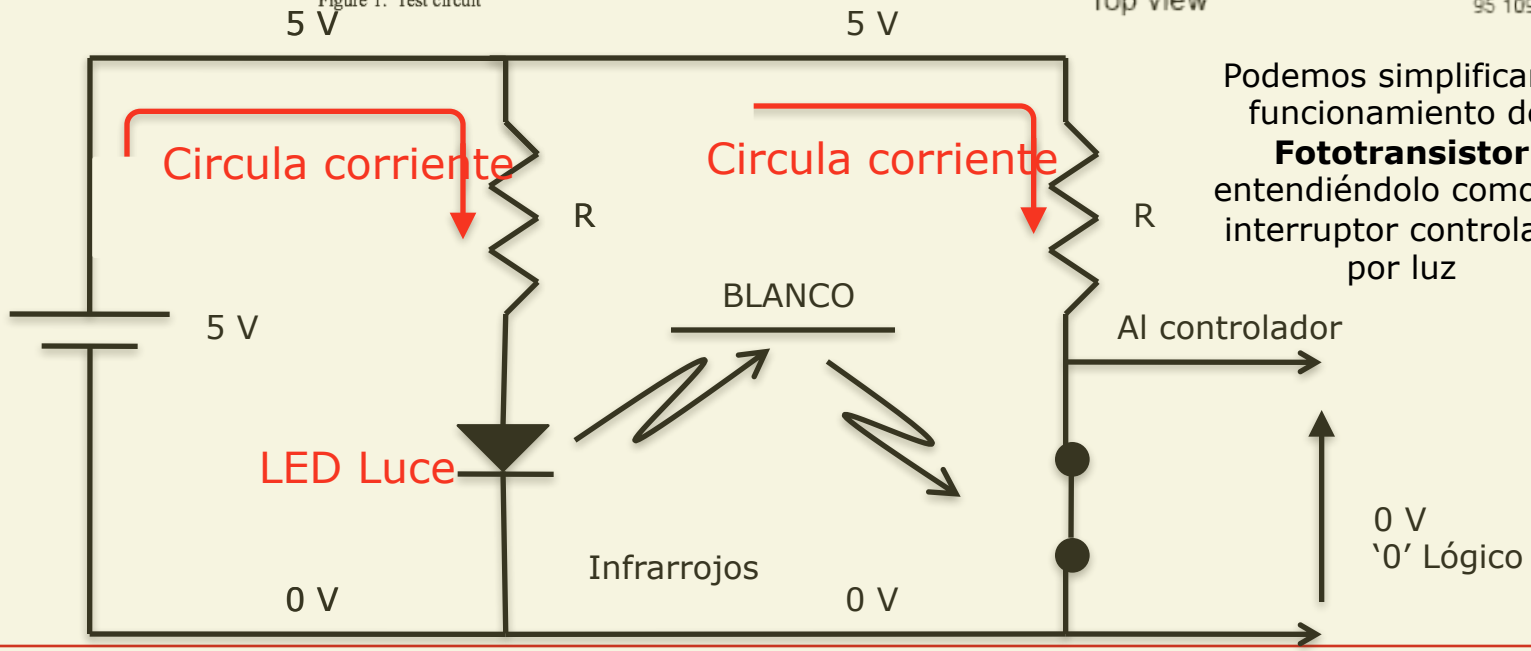


Figure 1. Test circuit





Podemos simplificar el funcionamiento del **Fototransistor** entendiéndolo como un interruptor controlado por luz

Leyendo los sensores de infrarrojos

- Ejemplo: Lectura de los sensores de infrarrojo (InfraredSerial)
  - Como siempre el programa tiene tres partes

```
// Declaración de los pines de los sensores
int pinInfraredRight = A1;
int pinInfraredLeft =  A2;

// Declaración de la variables donde se guarda el valor de los sensores
int infraredRight;
int infraredLeft;
```

```
void setup(){
  Serial.begin(19200);  // Se configura la comunicacion serie a 19200 baudios
}
```

```
void loop(){
   delay(200);

  // Se lee el valor de los sensores.
  infraredRight = analogRead(pinInfraredRight);
  infraredLeft  = analogRead(pinInfraredLeft);

 // Se envía al ordenador el valor de los sensores
  Serial.println("SensorIzd      SensorDer  ");
  Serial.print(infraredLeft);
  Serial.print("    |      ");
  Serial.println(infraredRight);
}
```

Leyendo los sensores de infrarrojos

- Ejemplo: Lectura de los sensores de infrarrojo (InfraredSerial)
  - Como siempre el programa tiene tres partes

```
// Declaración de los pines de los sensores
int pinInfraredRight = A1;
int pinInfraredLeft =  A2;

// Declaración de la variables donde se guarda el valor de los sensores
int infraredRight;
int infraredLeft;
```

Pines Analógicos:
Pueden tomar cualquier valor analógico de 0V a 5V

```
void setup(){
  Serial.begin(19200);  // Se configura la comunicacion serie a 19200 baudios
}
```

```
void loop(){
   delay(200);

  // Se lee el valor de los sensores.
  infraredRight = analogRead(pinInfraredRight);
  infraredLeft  = analogRead(pinInfraredLeft);

 // Se envía al ordenador el valor de los sensores
  Serial.println("SensorIzd      SensorDer  ");
  Serial.print(infraredLeft);
  Serial.print("    |      ");
  Serial.println(infraredRight);
}
```

analogRead(pin):
Esta función lee el voltaje que hay en el **pin** analógico dado como parámetro. Devuelve un valor de 0 a 1023.

Leyendo los sensores de infrarrojos

- Ejemplo: Lectura de los sensores de infrarrojo (InfraredSerial)
  - Como siempre el programa tiene tres partes

```
// Declaración de los pines de los sensores
int pinInfraredRight = A1;
int pinInfraredLeft =  A2;

// Declaración de la variables donde se guarda el valor de los sensores
int infraredRight;
int infraredLeft;
```

```
void setup(){
  Serial.begin(19200); // Se confi
}
```

Comunicación Serie:
Configuración a 19200 baudios.

```
void loop(){
   delay(200);

  // Se lee el valor de los sensores.
  infraredRight = analogRead(pinInfraredRight);
  infraredLeft  = analogRead(pinInfraredLeft);

 // Se envía al ordenador el valor de los sensores
  Serial.println("SensorIzd      SensorDer  ");
  Serial.print(infraredLeft);
  Serial.print("    |     ");
  Serial.println(infraredRight);
}
```

Se envía por el puerto serre el valor de las variables.

Jugando con los sensores de infrarrojos

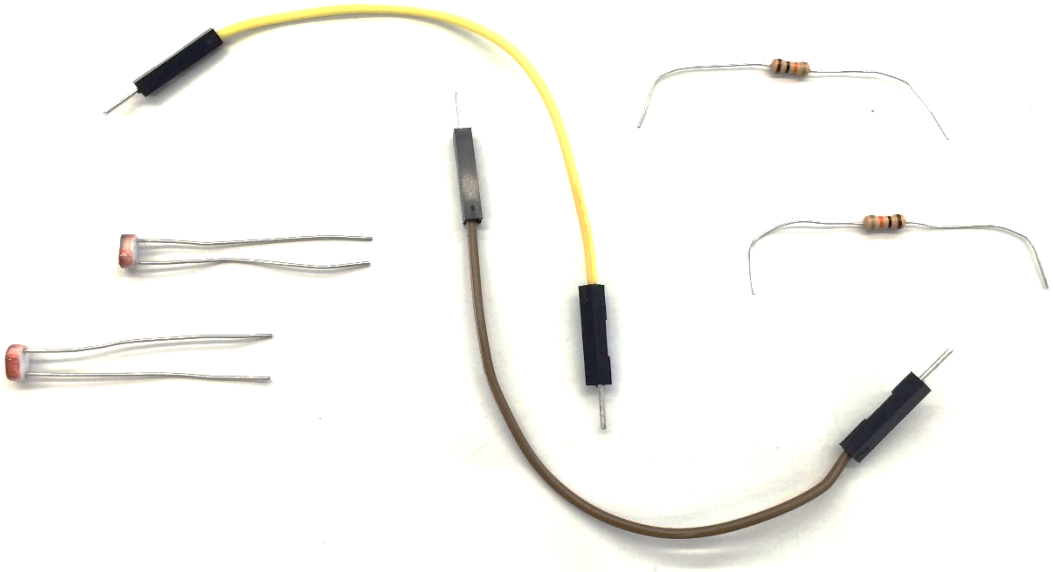
- **Ejercicios**:
  1. Realiza un programa que encienda un LED cuando detecta negro con un sensor y el otro LED cuando lo detecta con el otro sensor.
     - Para detectar el valor umbral para distinguir entre blanco y negro calcula, para cada sensor, el valor medio entre los valores de blanco y negro.
     - Puedes encontrar más información sobre la sentencia if en
       - https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C/Instrucciones_de_control
     - Escribe, verifica el programa y cárgalo en la placa
  2. Realiza un programa que encienda un LED cuando se detecta negro en los dos sensores.
     - Escribe, verifica el programa y cárgalo en la placa
  3. Realiza un programa que envíe por el puerto serie mensajes de texto como:
     - "Dos sensores en negro" "Dos sensores en blanco"
     - "Solo sensor de la derecha negro" "Solo sensor de la izquierda negro"

Jugando con los sensores de infrarrojos

- **Ejercicios**:
  4. Realiza un programa que permita calibrar automáticamente el umbral entre blanco y negro.
     - Al empezar, el robot debe ponerse con los dos sensores en blanco.
     - Al cabo de 1 segundo se enciende el LED
     - Se pone el robot con los dos sensores en negro
     - Se pulsa el pulsador
     - Se calcula automáticamente el umbral
     - A partir de ese momento se entra en loop() y se envía por el puerto serie la siguiente información:
       - Nivel detectado en los dos sensores
       - Umbral calculado
       - Resultado de la detección: blanco o negro
  - Escribe, verifica el programa y cárgalo en la placa

Conectamos los LDR

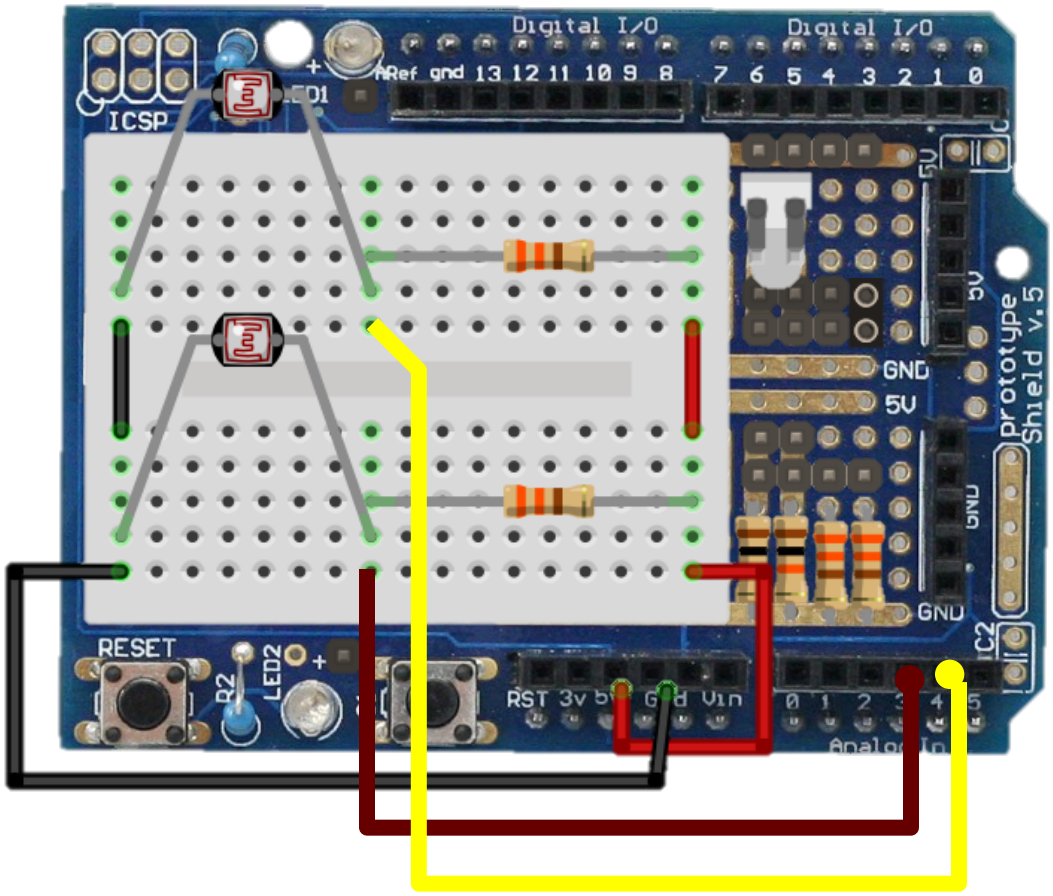
- Materiales



2 resistencias de 10kΩ
Marrón – **Negro** – **Naranja** - **Oro**

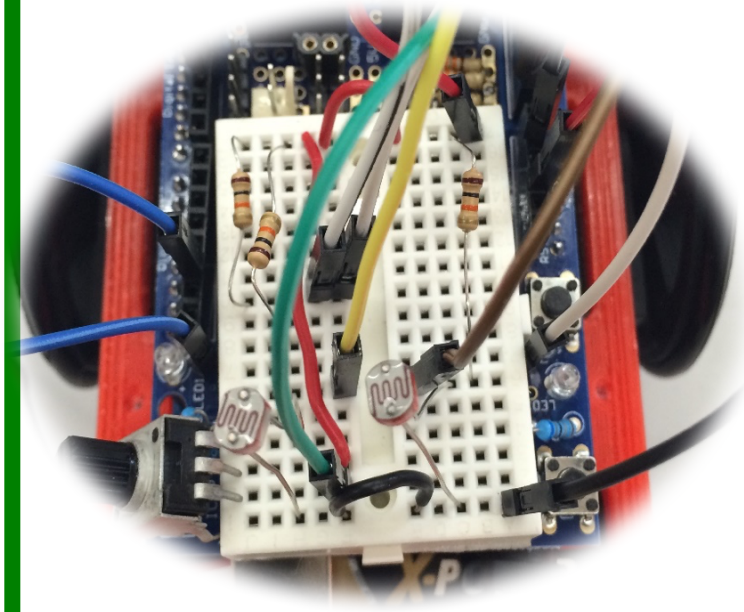
Conectamos los LDR

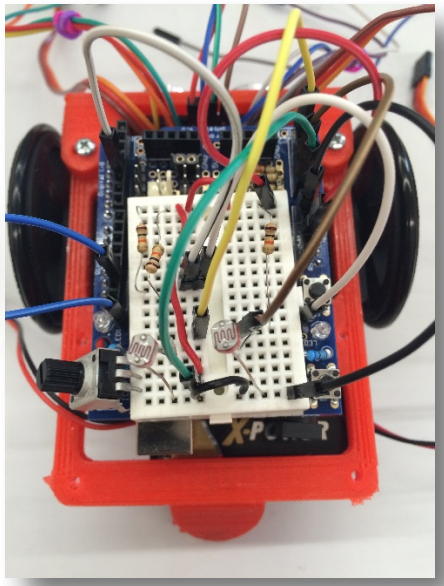
- Conecta las patillas del LDR como se indica en la figura.

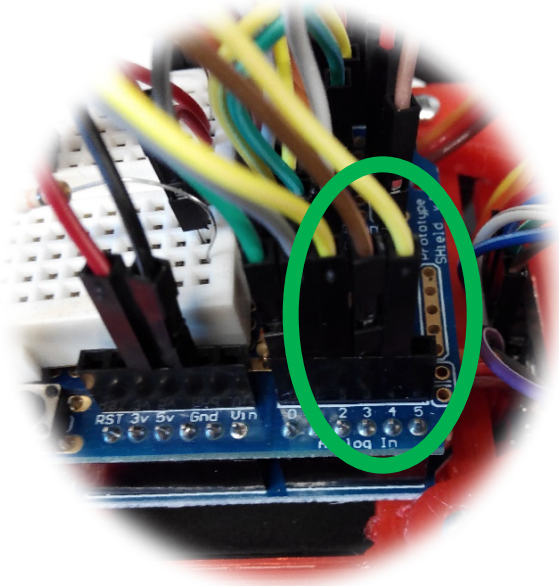


Conectamos los LDR

- Conecta las patillas del LDR como se indica en la figura.
  - Una patilla del LDR irá a Vcc y otra a un punto de interconexión de la protoboard. Realiza lo mismo para el otro LDR.
  - Conecta las resistencias de 10kOhm desde el punto de interconexión de la protoboard hasta GND.
  - Por último, conecta un cable desde las interconexiones entre el LDR y la resistencia hacia las entradas analógicas A3 y A4.







Leemos los LDR

- Ejemplo: Medida infrarrojos (LDR_example)
  - Medida de LDR.

```
// Pines de los sensores
int pinLDRRight =      A3;
int pinLDRLeft =       A4;
// Variables donde se guarda el valor de los sensores
int LDRRight;
int LDRLeft;

void setup(){
  Serial.begin(19200);  // Configuramos comunicacion serie
}
void loop(){
  delay(200);
  // Leemos el valor de los sensores
  LDRRight = analogRead(pinLDRRight);
  LDRLeft = analogRead(pinLDRLeft);

  // Imprimimos el valor de los sensores
  Serial.println("LDRIzd      LDRDer  ");
  Serial.print(LDRLeft);
  Serial.print("    |     ");
  Serial.println(LDRRight);
}
```

Jugando con las LDR

- **Ejercicios**:
  1. Realiza un programa que encienda un LED cuando las dos LDR detectan poca luz (“¡Encendemos las luces que se hace de noche!”)
     - Escribe, verifica el programa y cárgalo en la placa
  2. Realiza un programa que encienda un LED cuando las dos LDR detectan poca luz ajustando el umbral con el potenciómetro.
     - Escribe, verifica el programa y cárgalo en la placa
  3. Realiza un programa que detecte una luz potente delante del robot (por ejemplo la linterna del móvil) encendiendo un LED.
     - Escribe, verifica el programa y cárgalo en la placa

Recibiendo órdenes del ordenador

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)
  - Abre el ejemplo del botón que se llama ControlLedSerial.
    - Archivo->Ejemplos->Tubot2016Lib->1_Led->ControlLedSerial
  - Verifica y carga el programa en la tarjeta de control
  - Abre el Monitor Serie y configura la velocidad a 19200 baudios.
    - Escribe en la parte superior de la ventana una "**a**" y pulsa **Enviar**
      - **El LED debería encenderse**
    - Escribe en la parte superior de la ventana una "**s**" y pulsa **Enviar**
      - **El LED debería apagarse**

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed  =   13;     // Configura el pin donde está conectado el LED
int incomingByte = 0;   // Inicializa la variable donde se escribirán los datos recibidos

void setup(){
  Serial.begin(19200);     // Se inicializa la comunicacion serie a 19200 bits por segundo
  pinMode(pinLed, OUTPUT);  //CSe configura el pinLed como salida
}

void loop(){
    delay(200); //Esperamos 200 milisegundos

    // Comprueba si se ha recibido algún dato. Si no recibe nada Serial.available() devuelve cero
    if (Serial.available() > 0)
    {
       // Leemos la información que se recube desde el ordenador.
       incomingByte = Serial.read(); //Almacena el dato que recibe en una variable

        switch (incomingByte){  // Actuamos en función del valor de esta variable
           case 'a':          // Si hemos recibido el caracter 'a'
              digitalWrite(pinLed, HIGH); //Encendemos el LED
              break;

           case 's':          // Si hemos recibido el caracter 's'
              digitalWrite(pinLed, LOW); //Apagamos el LED
              break;
        }
    }
}
```

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed   =   13;     //
int incomingByte = 0;    //
```

Se declara el PIN asociado al LED
Y una variable donde se guardarán los datos recibidos

```
void setup(){
  Serial.begin(19200);     // Se inicializa la comunicacion serie a 19200 bits por segundo
  pinMode(pinLed, OUTPUT);  //CSe configura el pinLed como salida
}
```

```
void loop(){
    delay(200); //Esperamos 200 milisegundos

    // Comprueba si se ha recibido algún dato. Si no recibe nada Serial.available() devuelve cero
    if (Serial.available() > 0)
    {
       // Leemos la información que se recube desde el ordenador.
       incomingByte = Serial.read(); //Almacena el dato que recibe en una variable

        switch (incomingByte){  // Actuamos en función del valor de esta variable
           case 'a':          // Si hemos recibido el caracter 'a'
              digitalWrite(pinLed, HIGH); //Encendemos el LED
              break;

           case 's':          // Si hemos recibido el caracter 's'
              digitalWrite(pinLed, LOW); //Apagamos el LED
              break;
        }
    }
}
```

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed  =   13;     // Configura el pin donde está conectado el LED
int incomingByte = 0;   // Inicializa la variable donde se escribirán los datos recibidos

void setup(){
  Serial.begin(19200);     // Se in
  pinMode(pinLed, OUTPUT);  // Se c
}

void loop(){
    delay(200); //Esperamos 200 milisegundos

    // Comprueba si se ha recibido algún dato. Si no recibe nada Serial.available() devuelve cero
    if (Serial.available() > 0)
    {
       // Leemos la información que se recube desde el ordenador.
       incomingByte = Serial.read(); //Almacena el dato que recibe en una variable

        switch (incomingByte){  // Actuamos en función del valor de esta variable
           case 'a':          // Si hemos recibido el caracter 'a'
              digitalWrite(pinLed, HIGH); //Encendemos el LED
              break;

           case 's':          // Si hemos recibido el caracter 's'
              digitalWrite(pinLed, LOW); //Apagamos el LED
              break;
        }
    }
}
```

Se configura la comunicación serie a 19200 baudios
Se configura como salida el pin del LED

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed  =   13;     // Configura el pin donde está conectado el LED
int incomingByte = 0;   // Inicializa la variable donde se escribirán los datos recibidos

void setup(){
  Serial.begin(19200);     // Se inicializa la comunicacion serie a 19200 bits por segundo
  pinMode(pinLed, OUTPUT);  //CSe configura el pinLed como salida
}

void loop(){
    delay(200); //Esperamos 200 milisegundos

    // Comprueba si se ha recibido algún dato. Si no
    if (Serial.available() > 0)
    {
       // Leemos la información que se recube desde e
       incomingByte = Serial.read(); //Almacena el dato que recibe en una variable

        switch (incomingByte){  // Actuamos en función del valor de esta variable
           case 'a':          // Si hemos recibido el caracter 'a'
              digitalWrite(pinLed, HIGH); //Encendemos el LED
              break;

           case 's':          // Si hemos recibido el caracter 's'
              digitalWrite(pinLed, LOW); //Apagamos el LED
              break;
        }
    }
}
```

Serial.available():
Esta función indica el número de datos que se han recibido y están disponibles para ser leídos. Si no se ha recibido nada vale cero.

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed  =   13;     // Configura el pin donde está conectado el LED
int incomingByte = 0;   // Inicializa la variable donde se escribirán los datos recibidos

void setup(){
  Serial.begin(19200);     // Se inicializa la comunicacion serie a 19200 bits por segundo
  pinMode(pinLed, OUTPUT);  //CSe configura el pinLed como salida
}

void loop(){
    delay(200); //Esperamos 200 milisegundos

    // Comprueba si se ha recibido algún dato. Si no recibe nada
    if (Serial.available() > 0)
    {
       // Leemos la información que se recube desde el ordenador
       incomingByte = Serial.read(); //Almacena el dato que recib

        switch (incomingByte){  // Actuamos en función del valor
           case 'a':          // Si hemos recibido el caracter 'a'
              digitalWrite(pinLed, HIGH); //Encendemos el LED
              break;

           case 's':          // Si hemos recibido el caracter 's'
              digitalWrite(pinLed, LOW); //Apagamos el LED
              break;
        }
    }
}
```

Serial.read ():
Esta función lee el dato recibido por el puerto serie.

En este caso almacena en una variable lo que recibe.

Recibiendo órdenes del ordenador

- Ejemplo: Control de un LED desde el teclado del ordenador (ControlLedSerial)

```
int pinLed  =   13;     // Configura el pin donde está conectado el LED
int incomingByte = 0;   // Inicializa la variable donde se escribirán los datos recibidos

void setup(){
  Serial.begin(19200);     // Se inicializa la comunicacion serie a 19200 bits por segundo
  pinMode(pinLed, OUTPUT);  //CSe configura el pinLed como salida
}

void loop(){
    delay(200); //Esperamos 200 milisegundo

    // Comprueba si se ha recibido algún da
    if (Serial.available() > 0)
    {
       // Leemos la información que se recu
       incomingByte = Serial.read(); //Alma

        switch (incomingByte){  // Actuamos
           case 'a':           // Si hemos r
              digitalWrite(pinLed, HIGH); /
              break;

           case 's':           // Si hemos r
              digitalWrite(pinLed, LOW); //
              break;
        }
    }
}
```

```
Switch (variable) {
  case valor1:
    ...
    break;
 case valor1:
    ...
    break;
}
```

Esta orden de control permite ejecutar unas instrucciones u otras dependiendo del valor de una variable.
En el ejemplo al recibir “a” se enciende el LED y al recibir “d” se apaga

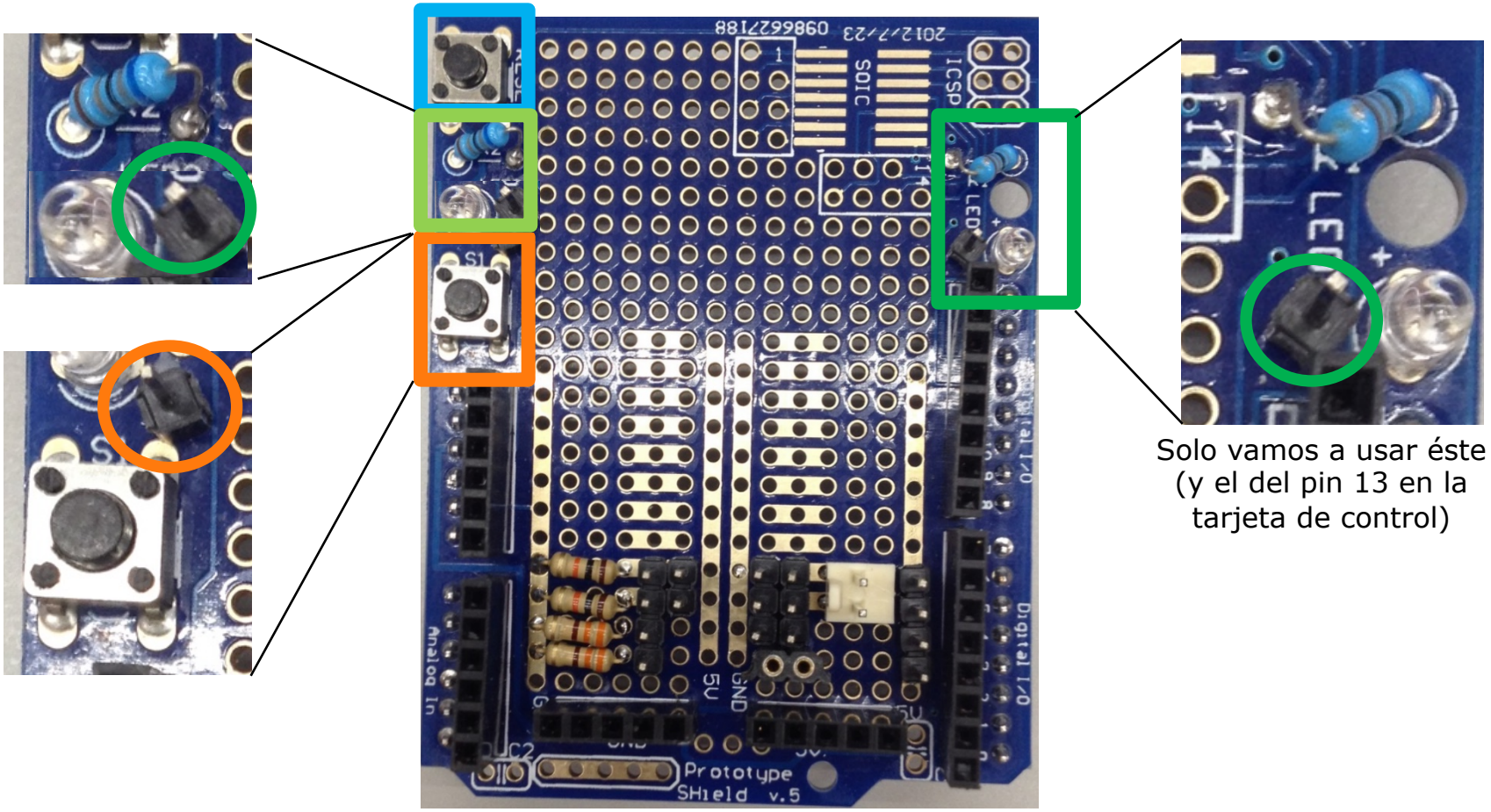
Puedes encontrar más información sobre la sentencia **switch** en https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C/Instrucciones_de_control

Recibiendo órdenes del ordenador

- **Ejercicios**:
  1. Modifica el programa para que además de encender o apagar el led envíe los mansajes:
     - "LED encendido" o "LED apagado"
  - Realiza un programa que encienda el LED cuando detecta un objeto más cerca de una distancia umbral. Haz que la distancia umbral pueda modificarse al enviar los códigos de las teclas "+" y "-". El programa debería enviar de vuelta el valor configurado.

Conexiones de la tarjeta de conexiones

- En la tarjeta está integrado el **Reset**, dos **LEDs** y un **pulsador**



Solo vamos a usar éste (y el del pin 13 en la tarjeta de control)

Conexiones de la tarjeta de conexiones

- Detalle de todas las conexiones de la tarjeta.